

A Reality of “Grid” Computing

–SamGrid–

Adam Lyon
(Fermilab Computing Division and DØ Experiment)
GridKa School'04
September, 2004

Outline

- Introduction
- Use Cases
- Deployment & Usage
- Implementation
- Operations, Monitoring, & Testing
- The Future

Data at an HEP Experiment



Detector (DØ)



Tape Storage



Compute Farm

- ◆ Collect data
- ◆ Reconstruct
- ◆ Skim

- ◆ Analyze
- ◆ Re-reconstruct
- ◆ Produce Monte Carlo

Then and now

- ◆ For Run I at DØ [1991–1997]:
 - ❖ Collected about 200 pb⁻¹ of data
 - ❖ Amounted to 60 TB total (all forms of data)
 - ❖ “Thumbnail” version of entire data lived on disk
 - ❖ Almost all processing was done at Fermilab
- ◆ For Run II at DØ [2000-]:
 - ❖ We have collected 470 pb⁻¹ so far (hope to get 4-8 fb⁻¹ by the end of the run)
 - ❖ We collect ~1 TB of raw data per day
 - ❖ We have saved 0.75 Petabytes to tape (expect 10-20+ PB)
 - ❖ Need to do re-reconstruction and analyses at remote locations
 - ❖ DØ reads the equivalent of Run 1 data every 11 days and writes Run 1 every 2 months

What do we need?

- ◆ Enormous amounts of data need to be transferred for different activities (**scalable**)
 - ◆ ... sometimes over large distances and with non-fault tolerant hardware (**robust**)
 - ◆ Knowledge of what we are doing and what we did (**monitoring and bookkeeping**)
 - ◆ Use our limited resources effectively both at home and away (**efficient**)
 - ◆ Don't want to know the details [where files sit, where jobs run] (**transparent**)
 - ◆ Find data easily (**query tools**)
-

Solution...

- ◆ An integrated data handling and job management system
- ◆ A GRID
- ◆ **SamGrid**
- ◆ SamGrid = SAM + JIM

What can SamGrid do?

- ◆ **SAMGrid manages file storage (replica catalogs)**
 - ❖ Data files are stored in tape systems at Fermilab and elsewhere. Files are cached around the world for fast access
- ◆ **SAMGrid manages file delivery**
 - ❖ Users at Fermilab and remote sites retrieve files out of file storage. SAMGrid handles caching for efficiency
 - ❖ You don't care about file locations
- ◆ **SAMGrid manages file metadata cataloging**
 - ❖ SAMGrid DB holds metadata for each file. You don't need to know the file names to get data
- ◆ **SAMGrid manages analysis bookkeeping**
 - ❖ SAMGrid remembers what files you ran over, what files you processed successfully, what applications you ran, when you ran them and where
- ◆ **SAMGrid manages jobs**
 - ❖ Choose execution site, deliver job and its needed data, store output

SamGrid Buzzword Glossary

A *project* runs on a *station* and requests delivery of a *dataset snapshot* to one or more *processes* on that station.

◆ **Project:** Run an application over data

◆ **Station:**

- ❖ Has processing power
- ❖ Has disk cache
- ❖ Can connect to outside world (for file transfers and DB access)
- ❖ Examples: Linux analysis cluster at DØ, GridKa's farm

◆ **Dataset:** metadata description which is resolved through a catalog query to file list. Datasets are named. Examples: (syntax not exact)

❖ data_type physics and run_number 78904 and data_tier raw

❖ request_id 5879 and data_tier thumbnail

◆ **Snapshot:** The list of files that satisfy the Dataset query at a particular time (*e.g.* start of the project)

◆ **Process:** User application (one or many exe instances) Examples: script to copy files; reconstruction job

Sample Use Cases

- I. Add Raw Detector Data to SamGrid
- II. Process Unskimmed Collider Data
- III. Process Skimmed Collider Data
- IV. Process Missed/New Data
- V. Monte Carlo Production
- VI. Process Simulated Data

I. Add Raw Detector Data to SamGrid

- ◆ Raw data collected into files by online detector DAQ
- ◆ Online system creates metadata for files
 - ❖ Run #
 - ❖ Start time/end time
 - ❖ Event catalog (triggers)
 - ❖ Luminosity info
- ◆ Online SamGrid station system submits files to SamGrid
- ◆ SamGrid stores files onto permanent storage and saves metadata to database

II. Process Unskimmed Collider Data

- ◆ Reconstruct raw data (production)
- ◆ Process the direct output of production
 - ❖ Skimming
 - ❖ Re-reconstruction
- ◆ User defines **dataset** by describing files of interest (not listing file names) using SamGrid command-line or GUI
 - ❖ `data_tier thumbnail and version p14.06.01 and run_type physics and run_qual_group MUO and run_quality GOOD`
- ◆ User submits project to SamGrid station (two ways)
 1. User selects station and submits with experiment's tools
 2. User submits to SamGrid, SamGrid job management chooses station (execution site) and manages project

III. Process Skimmed Collider Data

- ◆ Someone (a Physics group, the Common Skimming Group, or an individual) has produced skimmed files
- ◆ They created a **dataset** that describes these files
- ◆ You...
 - ❖ Submit project using their dataset name OR
 - ❖ Create a new dataset based on theirs and adding additional constraints

`__set__ DiElectronSkim and run_number 168339`

- ◆ Submission is same

IV. Process Missed/New Data

- ◆ The set of files that satisfy the dataset query at a given time is a **snapshot** and is remembered with the SamGrid project information
- ◆ One can make new datasets with:
 - ❖ Files that satisfy a dataset but are newer than the snapshot (new since the last project ran)
 - ❖ Files that **should** have been processed by the original project but were not consumed

```
__set__ myDataSet minus  
    (project_name myProject and  
     consumed_status consumed and consumer lyon)
```

V. Monte Carlo Production

- ◆ Physics group submits a SamGrid *Request* for MC production, giving parameters. SamGrid assigns a *Request Id*.
- ◆ SamGrid chooses execution site
- ◆ Workflow manager (*Runjob*) oversees production (event generator, simulator, reconstruction)
- ◆ SamGrid launches job to merge output files and submit them into SamGrid catalog and storage

VI. Process Simulated Data

- ◆ Look up simulation request with parameters of interest
 - ❖ e.g. Request **5874** has top Monte Carlo generated using Pythia with $m_t = 174 \text{ GeV}/c^2$
- ◆ Define **dataset** (via command-line or GUI):
 - ❖ `request_id 5874` and `data_tier thumbnail`
- ◆ Submit project

SamGrid Deployment

◆ DØ

- ❖ SamGrid is *THE* data handling system. Has been in **production** for five years. 45 active SamGrid stations deployed worldwide (including GridKa)
- ❖ Moving to SamGrid's automated job management system (10 execution sites so far)

◆ CDF

- ❖ Completing testing and migration to SamGrid for data handling in production
- ❖ Large analysis station at FNAL, 8 major remote stations (Italy, GridKa, Taiwan, Toronto, ...)

◆ MINOS

- ❖ Initial deployment underway

◆ US-CMS

- ❖ Using SamGrid metadata catalog components for proof-of-principle

SamGrid Statistics (8/2003-8/2004)

File delivery and consumption

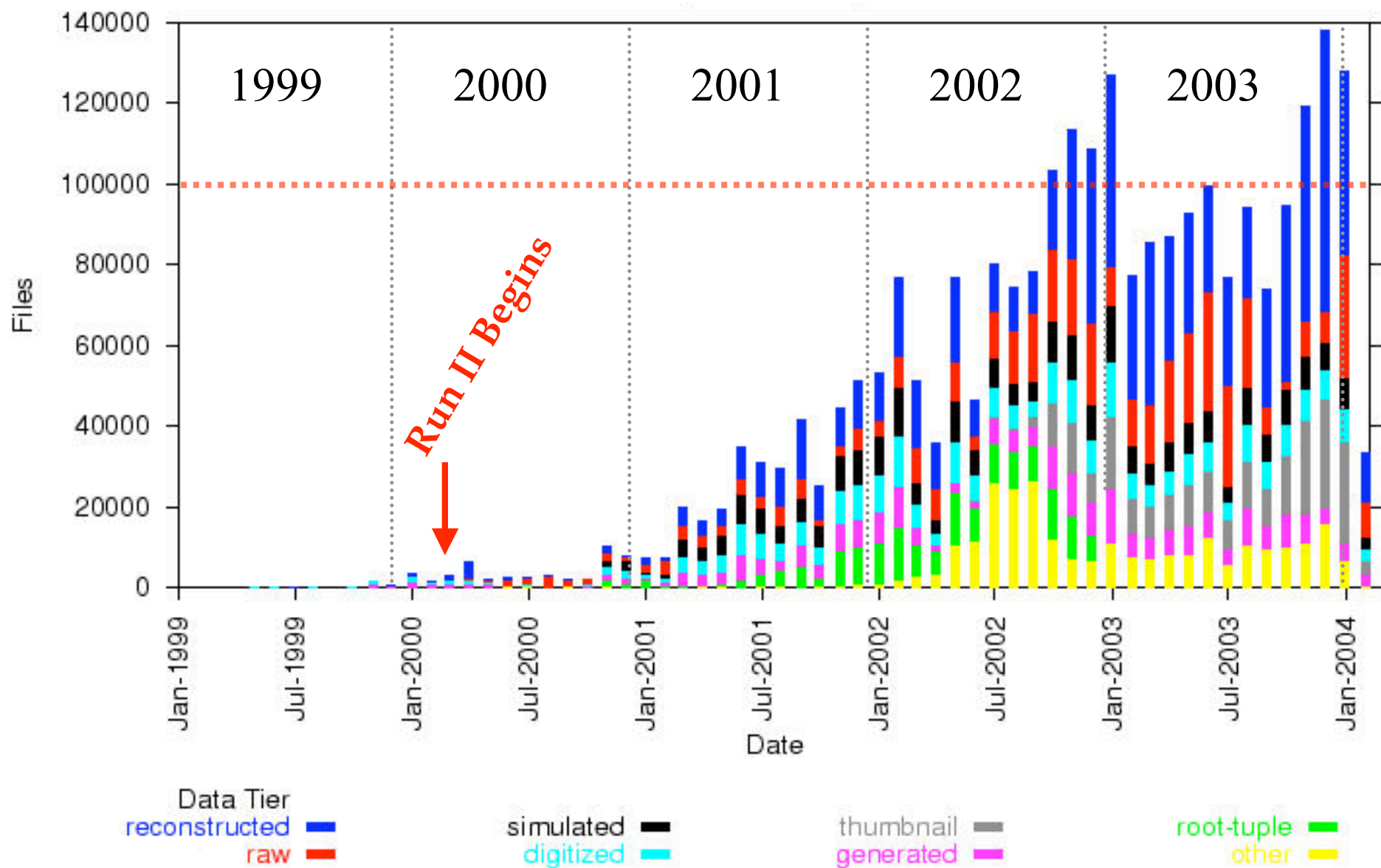
◆ DØ (production):

	# files (K)	Terabytes	# Events (B)
Total	4000	2000	48.0
Remote	500	142	3.8
GridKa	100	47	1.6

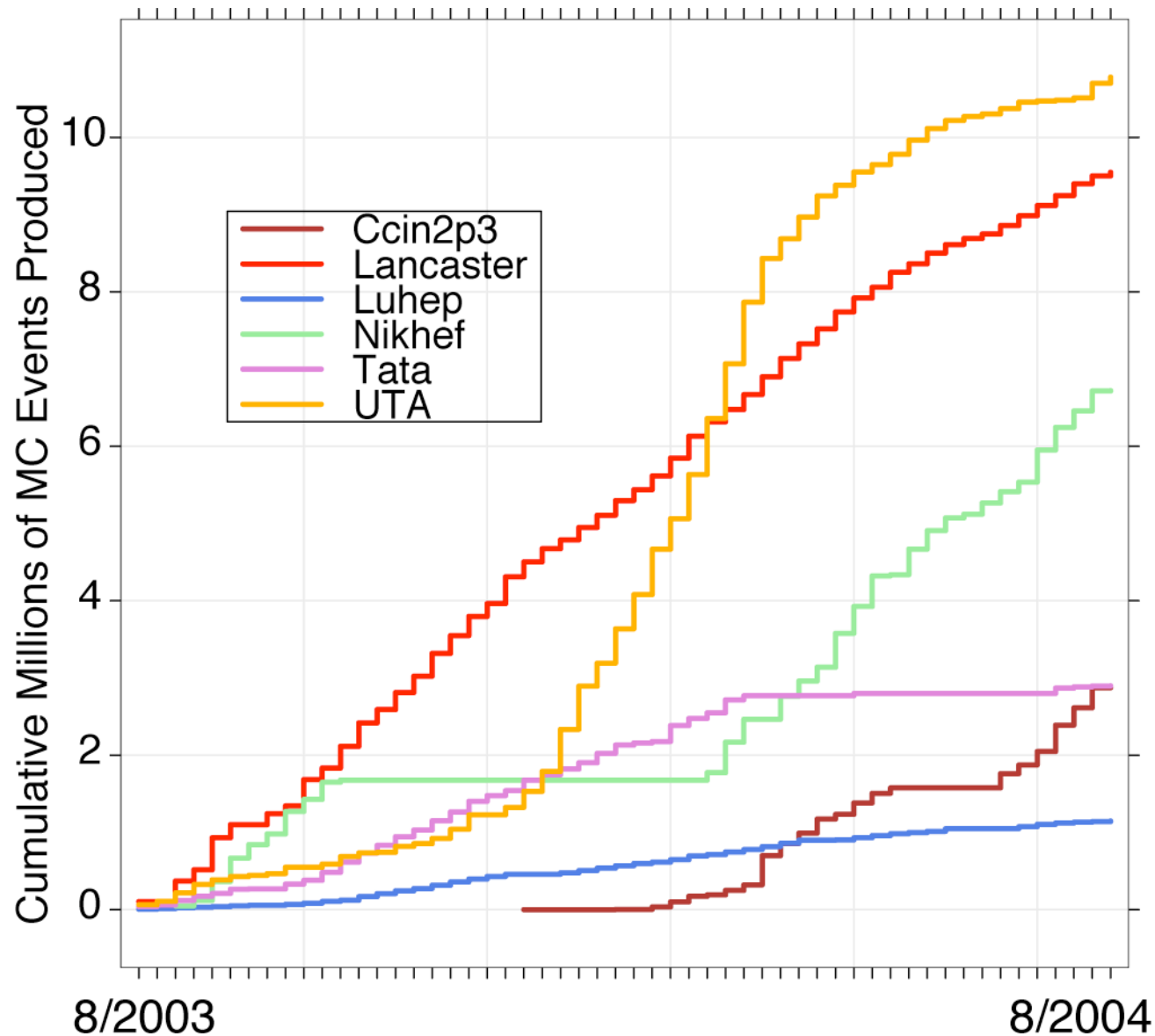
◆ CDF (testing and initial production):

- ❖ Total: 1.5 PB, 12B events
- ❖ GridKa largest offsite SAM consumer
- ❖ Can reach peak of 25 TB/day at FNAL

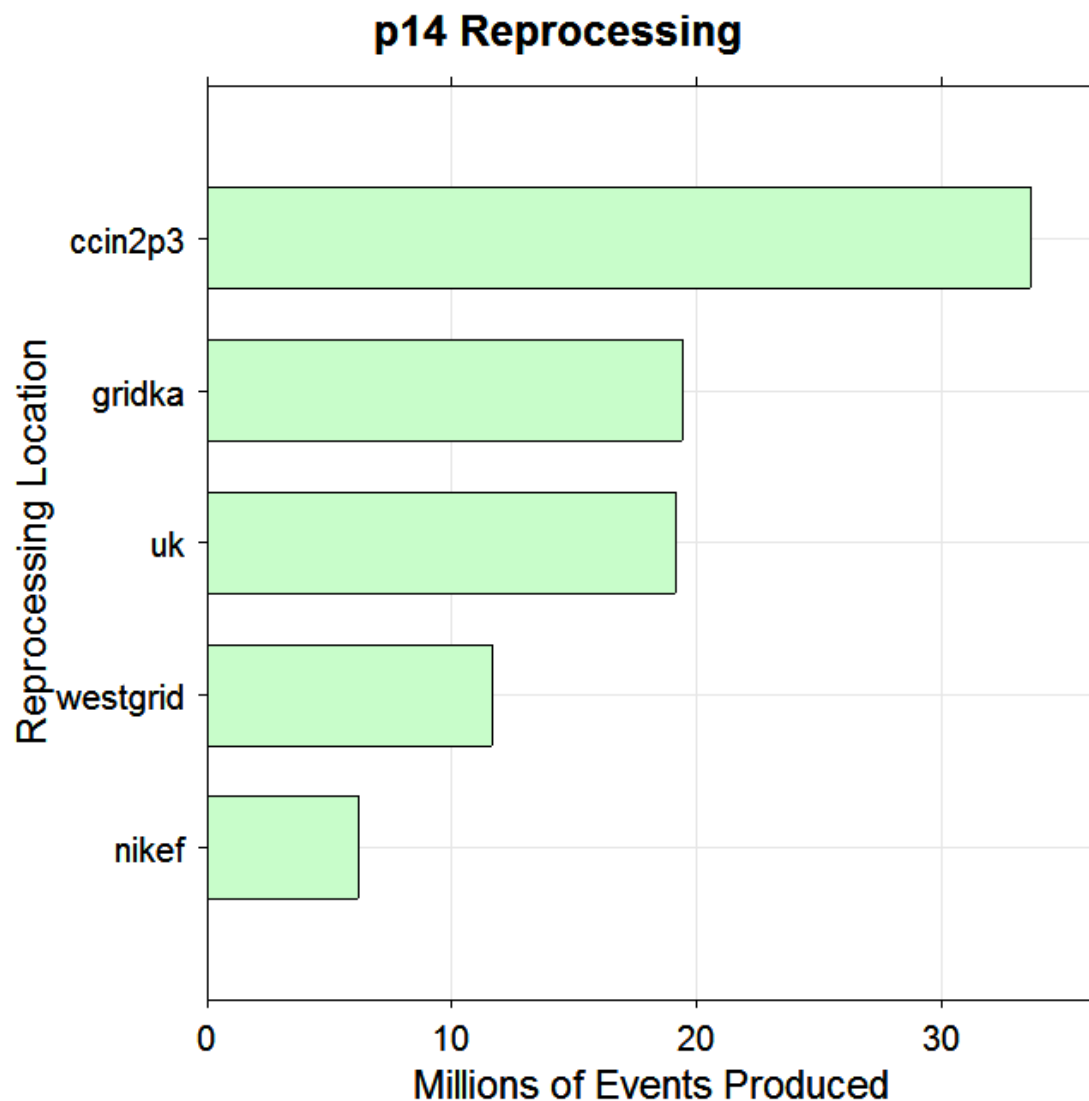
DØ SamGrid File Delivery (Files delivered by month)



DØ Monte Carlo Production (all remote)



DØ Past Re-reprocessing



Implementation of SamGrid

Overview

◆ Metadata

- ❖ Metadata is the conceptual glue for SamGrid
- ❖ Tight coupling

◆ Database

- ❖ Repository of metadata
- ❖ DBServers provide easy access

◆ Services

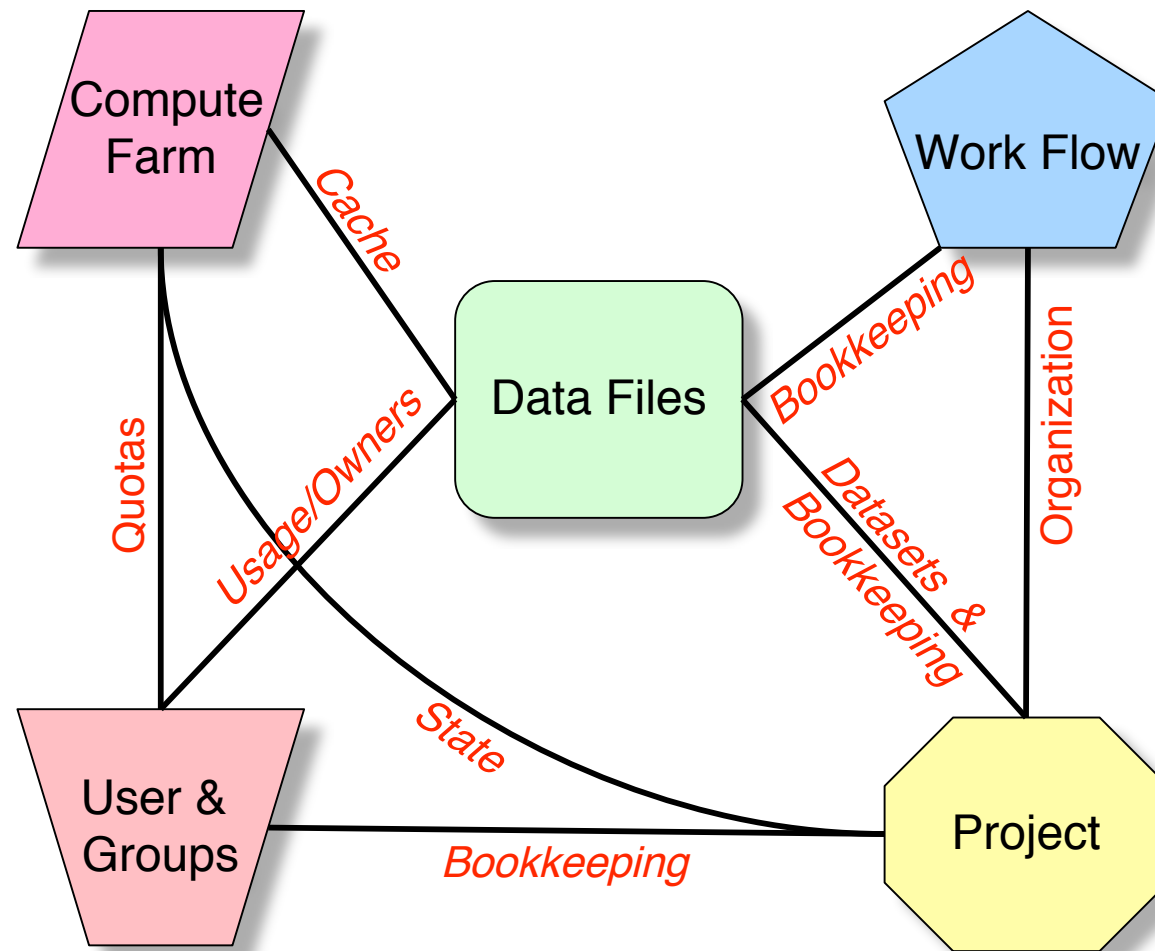
- ❖ Stations, stagers, workers, storage servers, submission sites, execution sites

◆ Client Side

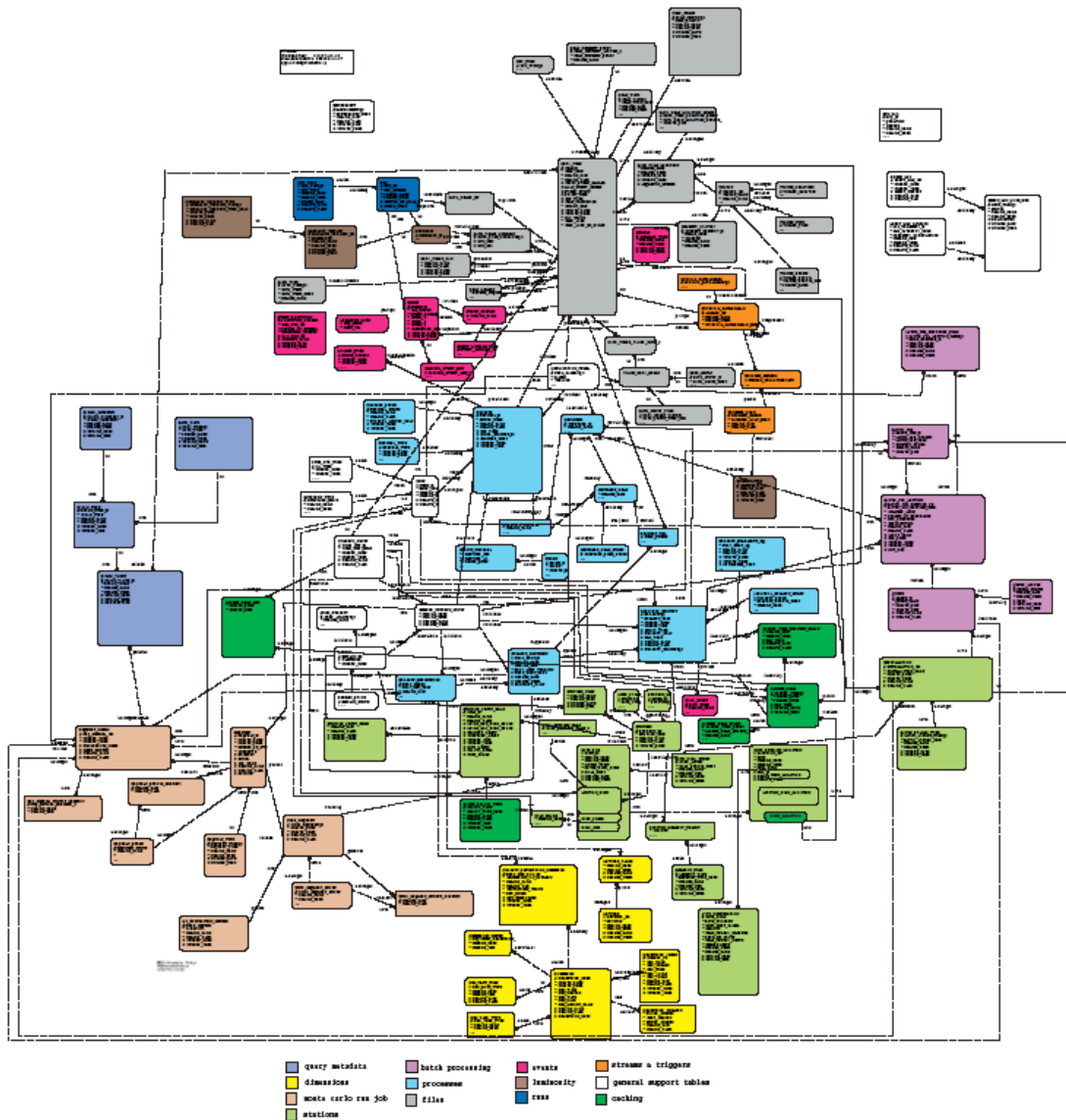
- ❖ The user experience

The Glue: Metadata

- ◆ “SamGrid is a collection of services each of which is described by metadata.” Metadata are **interrelated**.

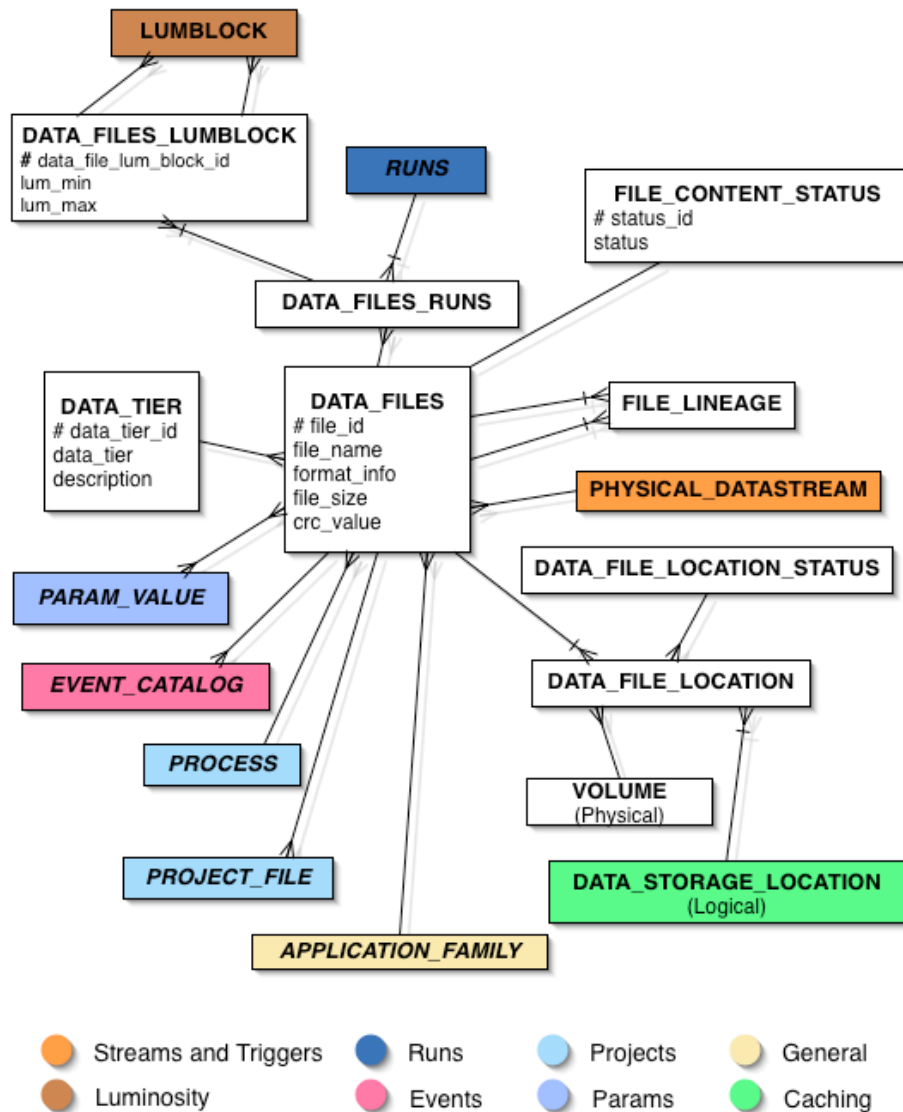


SamGrid Database



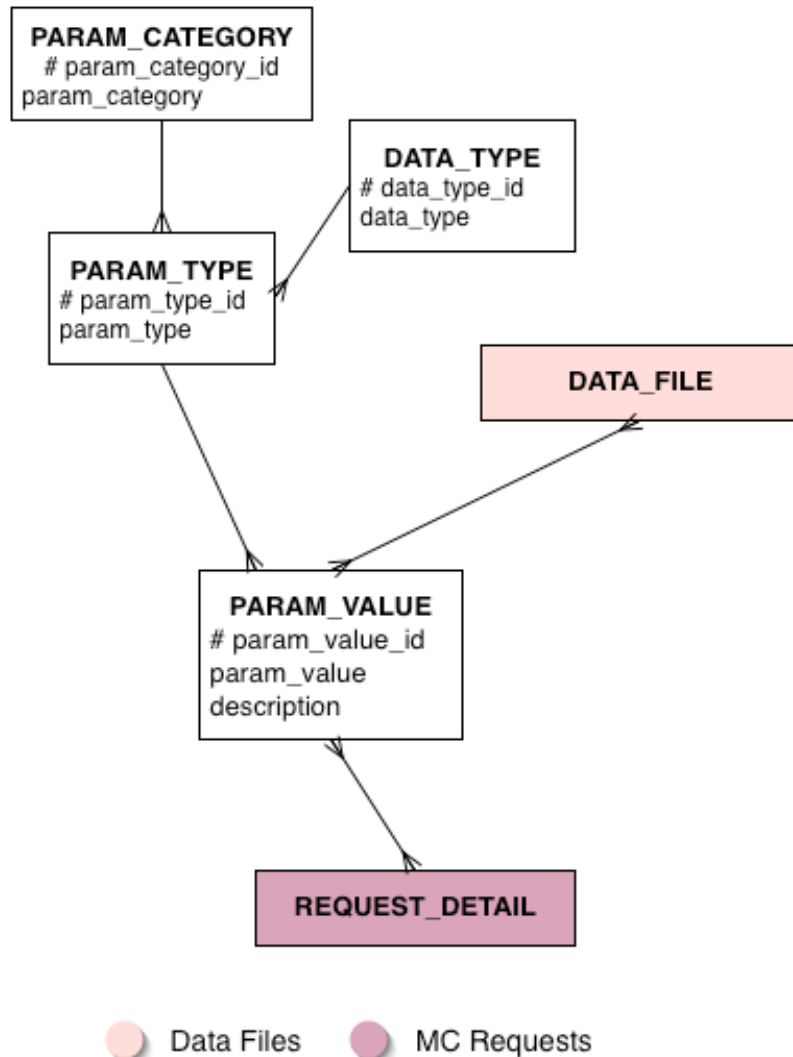
- ◆ DØ, CDF, and MINOS use the same DB Schema shown here
- ◆ Relational
 - ❖ Matches metadata
- ◆ Monolithic
 - ❖ Interrelated information are close by
- ◆ Flexible
 - ❖ Schema updates are allowed, but are carefully controlled
- ◆ Successful!
 - ❖ In production use at DØ for five years. *It may look scary, but it is well understood and it works!*

Data Files Metadata



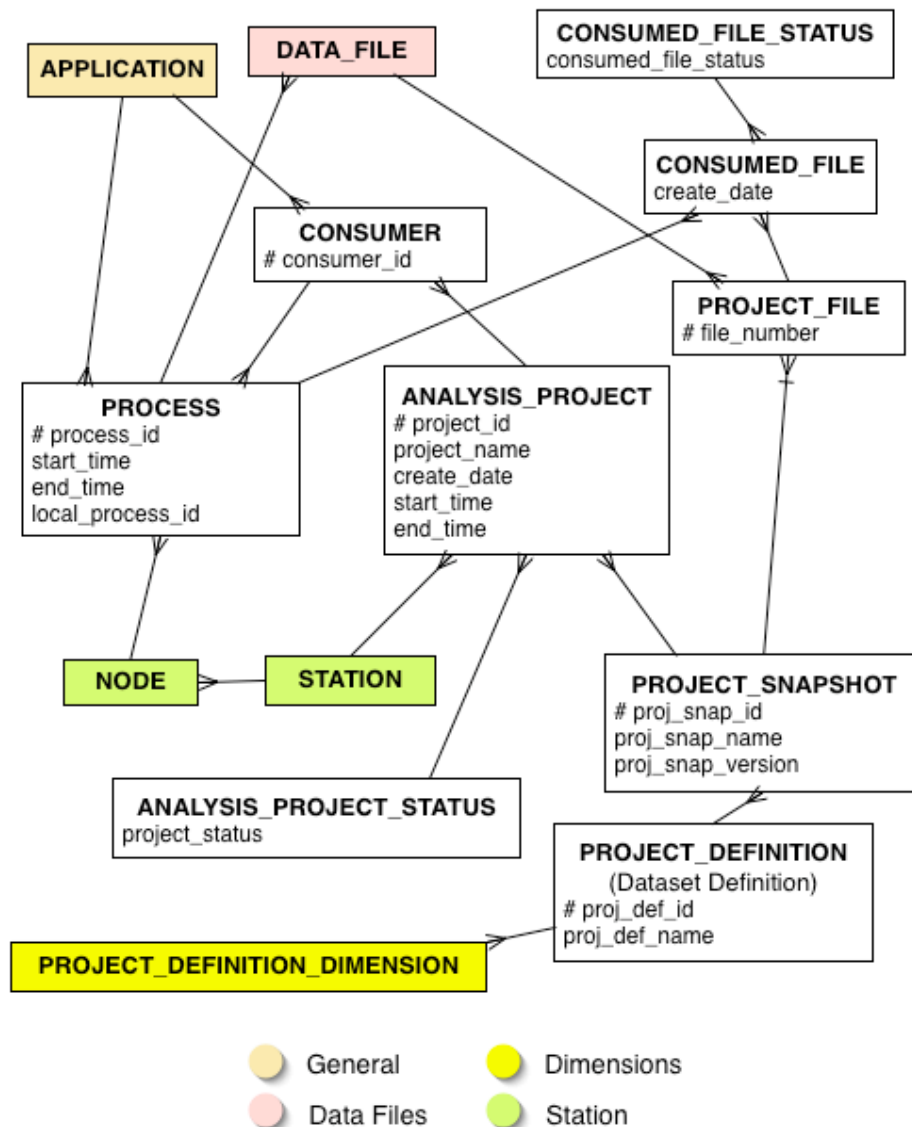
- ◆ Data Files: The heart of SamGrid
- ◆ Fixed metadata
 - ❖ File name, size, crc
 - ❖ Production group
 - ❖ Data Tier (Raw, Reconstructed, Thumbnail)
 - ❖ Application
 - ❖ Locations
 - ❖ Detector Runs
 - ❖ Event info
 - ❖ Project/Process
 - ❖ Luminosity
 - ❖ Stream/Trigger
- ◆ Connection to free metadata (Params) ...

Params (Free file metadata)



- ◆ Fixed metadata allows easy and performant querying
- ◆ Free metadata for application specific items
 - ❖ Categories group parameters (pythia, isajet, ...)
 - ❖ Types are the keywords (decayfile, topmass, ...)
 - ❖ Values
 - ❖ Queries are more difficult

Project Metadata



- ◆ Projects run on a dataset **Snapshot** with nodes from a SAMGrid station
- ◆ A Project has one or more **Consumers** (usually one)
- ◆ A Consumer has one or more **Processes**
- ◆ A Process is a job on a node. Keeps track of consumed files

Database Details

- ◆ Centralized Oracle Database at FNAL
- ◆ Three tier system ensures DB integrity (for all DBs at Fermilab)
 - ❖ Development - Newest schema with artificial or special data. Used for testing
 - ❖ Integration - Test new schema with replica of production data
 - ❖ Production - The real thing

Central vs. Distributed DB Design

◆ Pros of Central

- ❖ Database software easier to write, manage, and control
- ❖ DB queries are simpler and more performant

◆ Cons of Central

- ❖ Single point of failure - all data handling can stop
 - Hardware and network outages
 - Need to apply updates (DØ mitigates with monthly down day)
- ❖ Perhaps too monolithic (station must access DB to discover its cache disks)

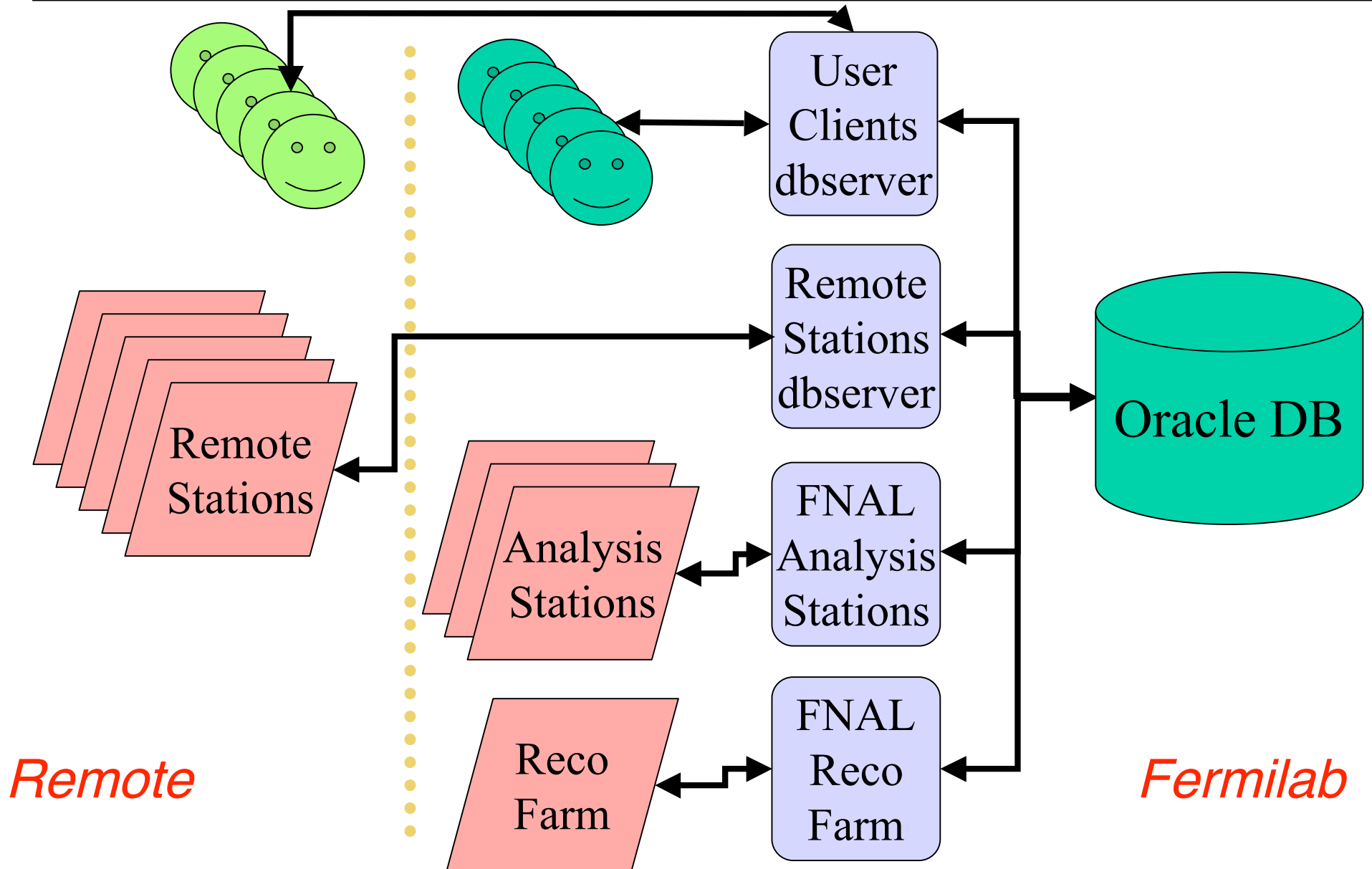
◆ Future directions

- ❖ Information servers to remotely cache DB information
- ❖ Initiative with a small business to produce software to transparently query distributed databases
- ❖ But I doubt we'll split off much of the metadata

DB Servers (Middleware)

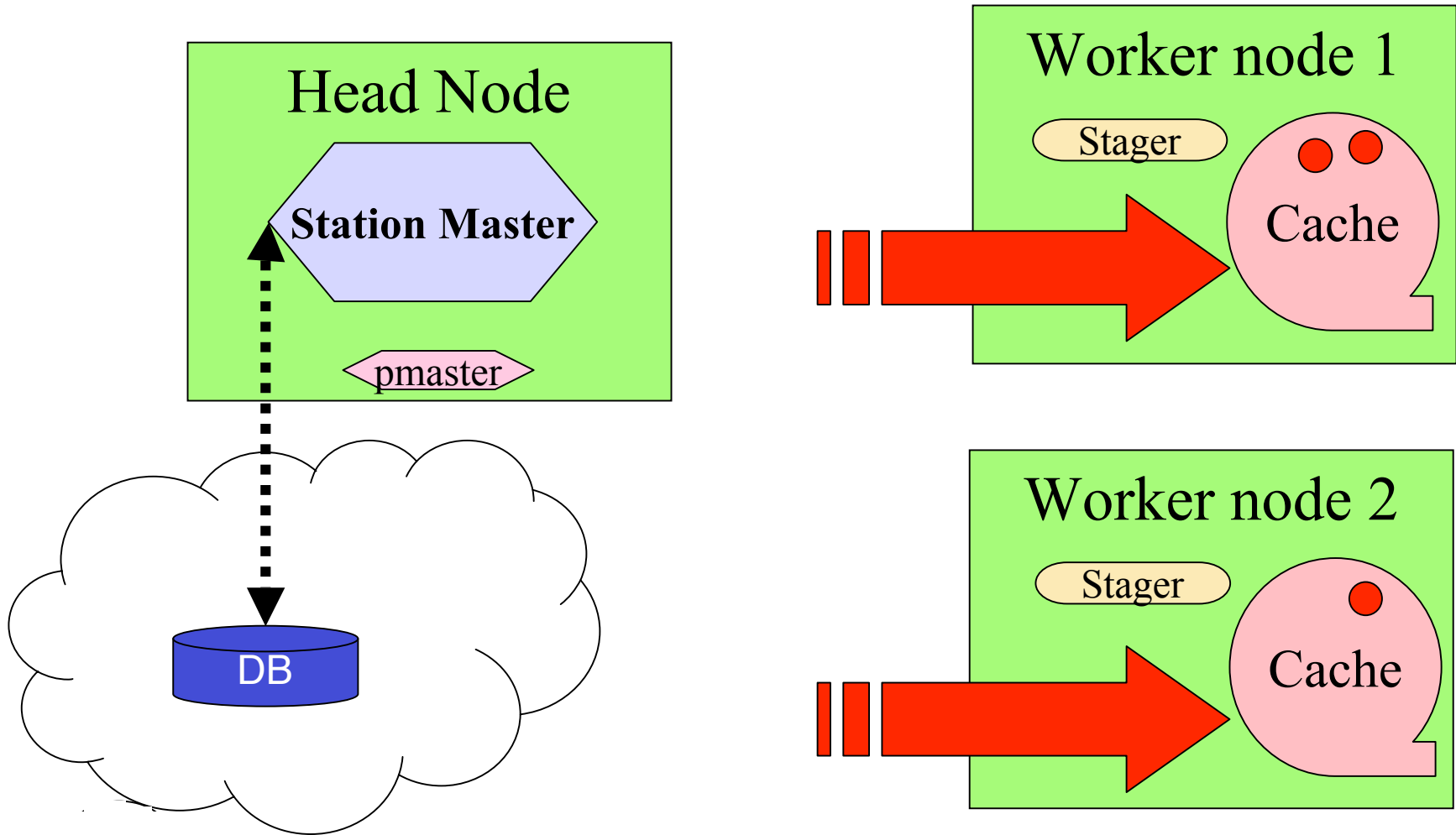
- ◆ Clients do not connect directly to Oracle but instead go through DB Server middleware
 - ❖ Use a CORBA Infrastructure (standardize DB access)
 - ❖ Server written in Python
 - ❖ Client interfaces with Python and C++
- ◆ DBServer Improvements
 - ❖ Multithreading
 - ❖ Revamped CORBA Infrastructure

DB Server Deployment



SamGrid Data Handling Services

Many station configurations are possible



SamGrid Data Handling Services

◆ Station Master

- ❖ Runs on head node, one instance, persistent, robust
- ❖ Coordinates file deliveries to compute farm
- ❖ Accesses the DB server

◆ Project Master

- ❖ Runs on head node (future distributed), one per project
- ❖ Coordinates file deliveries to running processes, tracks file consumption

◆ Stager

- ❖ Runs on node with cache to manage those disks
- ❖ Clears old files if room is needed
- ❖ Initiates file transfers (use *sam_cp*, wrapper for rcp, grid-ftp)

Special features of Data Handling

◆ Project can manage parallel processes

- ❖ Multiple processes (batch jobs) can pull files from the project's dataset
- ❖ Files spread among processes evenly
- ❖ If a process dies, others pick up the slack

◆ File delivery both optimized and throttled for performance

- ❖ SamGrid tries to deliver files **before** the jobs need them (prefetching)
 - File delivery can start before the processes start
 - File delivery continues while processes are executing
 - On FNAL analysis farm, 40% of time process did not need to wait for file
- ❖ Can set limits on simultaneous transfers
 - Avoids overloading network

◆ Files may come from multiple sources and different transports

- ❖ Sources are tape systems (FNAL enstore), other stations, other worker cache disks
- ❖ Transfers via grid-ftp, kerberized rcp, AFS, ... (wrap with *sam_cp*)

Job Information & Management

◆ Client “site”:

- ❖ User writes JDL and submits job to SamGrid
- ❖ User closes laptop (laptop only needs submission client software) and gets on plane

◆ Submission Site:

- ❖ Submission site calls on broker to determine execution site (criteria: load, files in cache, ...)
- ❖ (Execution sites advertise classads, and connect with SamGrid catalog)
- ❖ Submission site transfers job to execution site, job(s) enter local batch system

Job Information & Management

◆ Execution site:

- ❖ Submission site transfers bootstrap sandbox, is unpacked on head node
- ❖ Jobs awaken, SamGrid transfers needed software to node (samClient allows for SamGrid use on vanilla nodes)
- ❖ Jobs request data files from SamGrid and run
- ❖ Result files stored back into SamGrid. Log files sent back to submission site

◆ Client

- ❖ User lands, opens laptop, retrieves logs from submission site, gets result files out of SamGrid, discovers something new!

Job Management Details

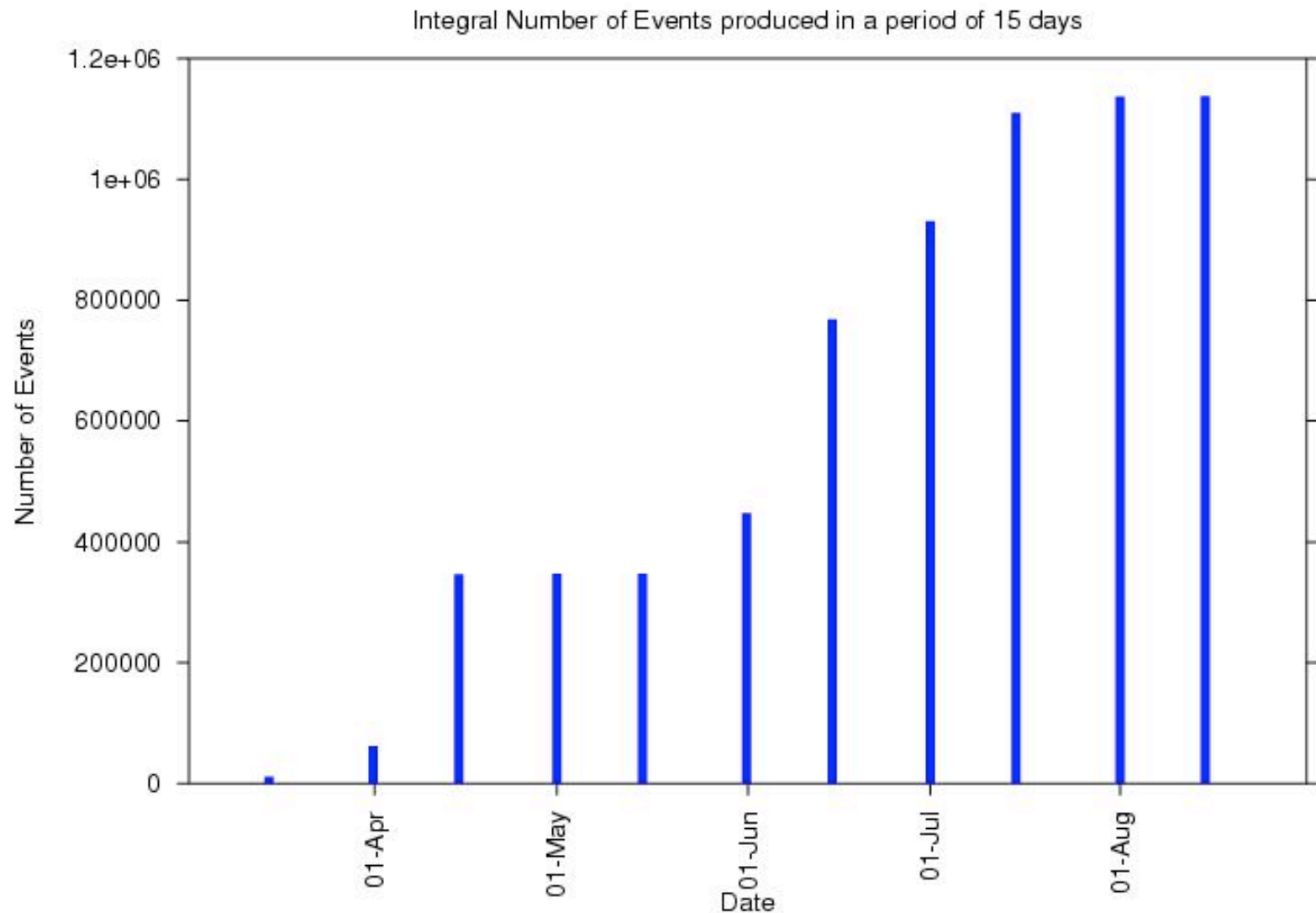
◆ Grid (sites talking to each other)

- ❖ Control, monitor, and transfer of information between sites
- ❖ Uses standard grid tools (Globus: gridftp, gram, mds) and Condor-g

◆ Fabric (collection of services and resources on site)

- ❖ Turned out managing the fabric was the real work for DØ
- ❖ Sandboxing, job driving, workflow, setting up application
- ❖ SamGrid uses a thick interface to weave the fabric (needs knowledge of application, batch system, ...)
- ❖ Thick interface can determine job status, even if job is sleeping - useful for monitoring
- ❖ Perhaps this should have been experiment's responsibility, but...

Monte Carlo Production via SamGrid Automated Job Management




User Experience

- ◆ Command line tools to query SamGrid services
 - ❖ `sam translate constraints --dim="data_tier thumbnail"`
- ◆ Dimension language to shield users from SQL
 - ❖ Extensible, Improving
- ◆ Web interfaces
 - ❖ DB queries
 - ❖ Dataset creation
- ◆ Command line administrative tools

Operations, Monitoring & Testing

- ◆ SamGrid shifters watch the system and respond to users' questions/requests
 - ❖ Cover 18 hours per day
 - ❖ Shifters in US, Canada, Europe, India, Brazil
- ◆ SamGrid experts at Fermilab rotate pager
- ◆ Local site SamGrid admins too
- ◆ Many tailored tools for monitoring
- ◆ Shifters and close monitoring beget much good will from users

Sam-At-A-Glance




SAM At A Glance

















D0 Production Environment

This page generated on 16 Sep 2004 13:01:28

Other Sam Diagnostics



SAMStations:


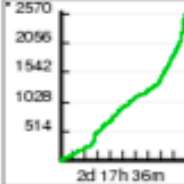

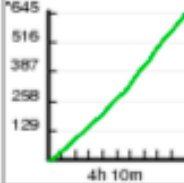

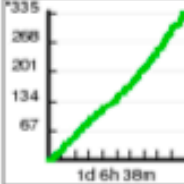

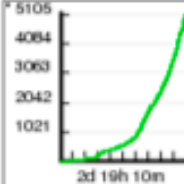
Monitor Level: Critical		Host:Port	Version	Up Since
	datalogger-d0olb	d0olb-gb-1.fnal.gov:1728	v4_2_1_54	13 Sep 2004 09:21:01
	datalogger-d0olc	d0olc-gb-1.fnal.gov:3066	v4_2_1_54	13 Sep 2004 09:19:54
Monitor Level: High		Host:Port	Version	Up Since
	central-analysis	d0mino.fnal.gov:43516	v4_2_1_76	14 Sep 2004 10:24:44
	fnal-farm	d0bbin.fnal.gov:38851	v4_2_1_61	09 Aug 2004 21:05:23
	cab	d0mino.fnal.gov:1609	v4_2_1_64	09 Sep 2004 15:18:05
	clued0	flotsam-clued0.fnal.gov:34206	v4_2_1_72	13 Sep 2004 09:13:26
	fnal-cabsrv1	d0srv001.fnal.gov:45823	v4_2_1_77	16 Sep 2004 10:54:31
Monitor Level: Normal		Host:Port	Version	Up Since
	cab-test	d0cs001.fnal.gov:36993	unknown	unreachable
	ccin2p3-analysis	ccd0.in2p3.fr:4501	v4_2_1_64	10 Sep 2004 10:34:16
	central-router	d0mino.fnal.gov:1643	v4_2_1_76	09 Sep 2004 15:18:30
	cinvestav-station			
	common-archive	d0mino.fnal.gov:1623		
	d0-fsuhep	lv6_0_1_1_no_smaster		09 Sep 2004 15:18:19
	d0-umich	lnxc25.hep.fsu.edu:58106	v4_2_1_65	16 Aug 2004 13:10:58
	d0_fzu_prague	dzero2.engin.umich.edu:32772	unknown	unknown
	d0ift	sam.farm.particle.cz:60343	v4_2_1_56	15 Sep 2004 04:57:35

SamTV (DØ)

samTV - Sam Snapshot Summaries

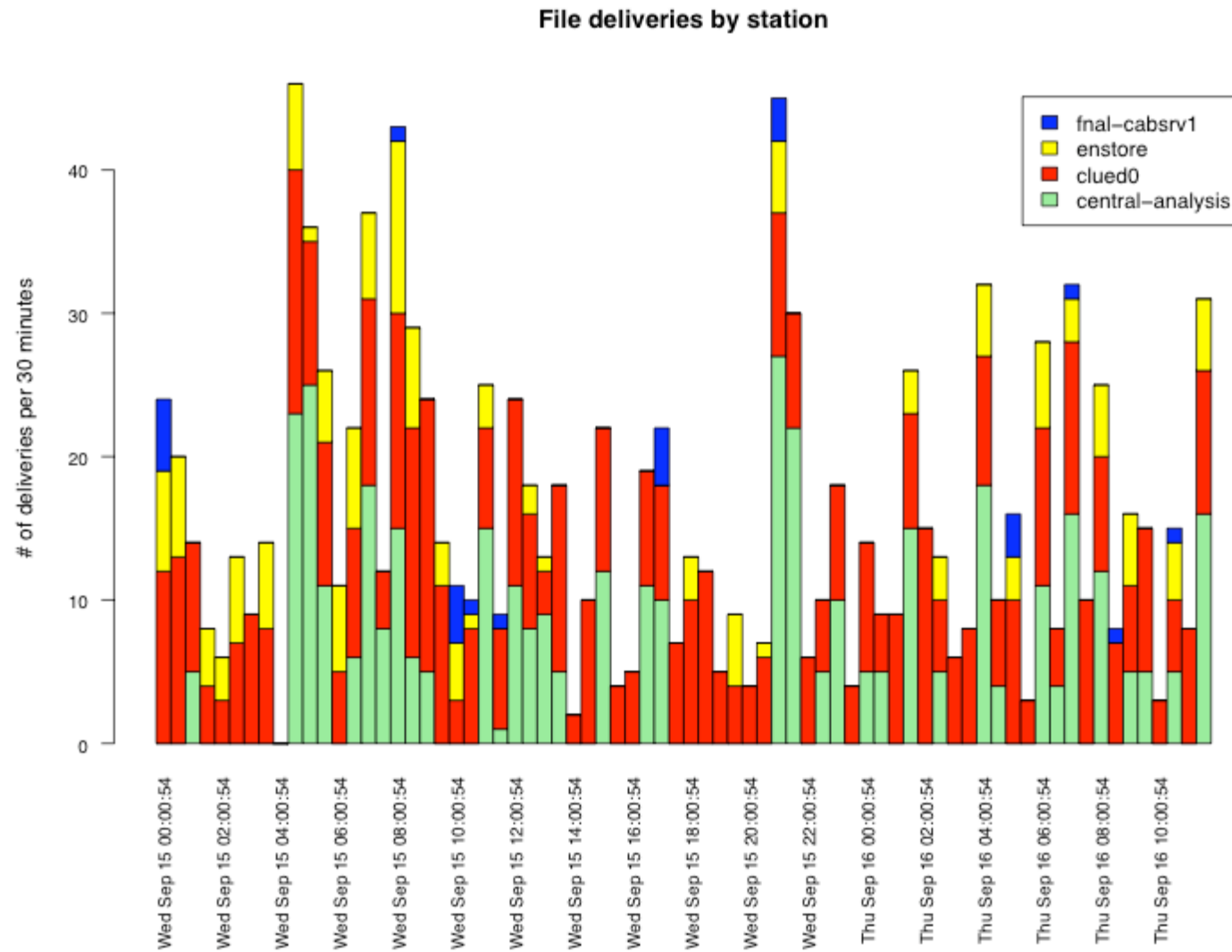
Produced on Thu Sep 16 12:44:41 2004

Jump to current [Station Histories](#) | Jump to [Old Summaries](#)

Station	Snapshot Create Time	Requested Files	Projects (tot run)	Projects Health (ok, error, waiting)	Last File Delivery	Deliveries
cab (history)	Thu Sep 16 12:33:32 2004	1000	28 18		Thu Sep 16 12:32:58 2004 (34s) CSskim-2EM-p14.06.00_post1-mh150904.job2	
central-analysis (history)	Thu Sep 16 12:35:22 2004	502	8 8		Thu Sep 16 12:35:09 2004 (13s) dalton-pick_event-08-19-46-16Sep2004	
central-router (history)	Thu Sep 16 12:37:13 2004	51	0 0	---	---	---
clued0 (history)	Thu Sep 16 12:33:01 2004	80	17 17		Thu Sep 16 12:26:44 2004 (6m 17s) bandurin_20040914163010	
fnal-cabsrv1 (history)	Thu Sep 16 12:37:36 2004	93	51 45		Thu Sep 16 12:37:33 2004 (3s) jdegenha-15Sep2004142738	

- ◆ Quickly check health of projects on FNAL stations
- ◆ Can discover if a station is having delivery problems
- ◆ Users can check on the status of their projects

SamTV History



Job Management Monitoring

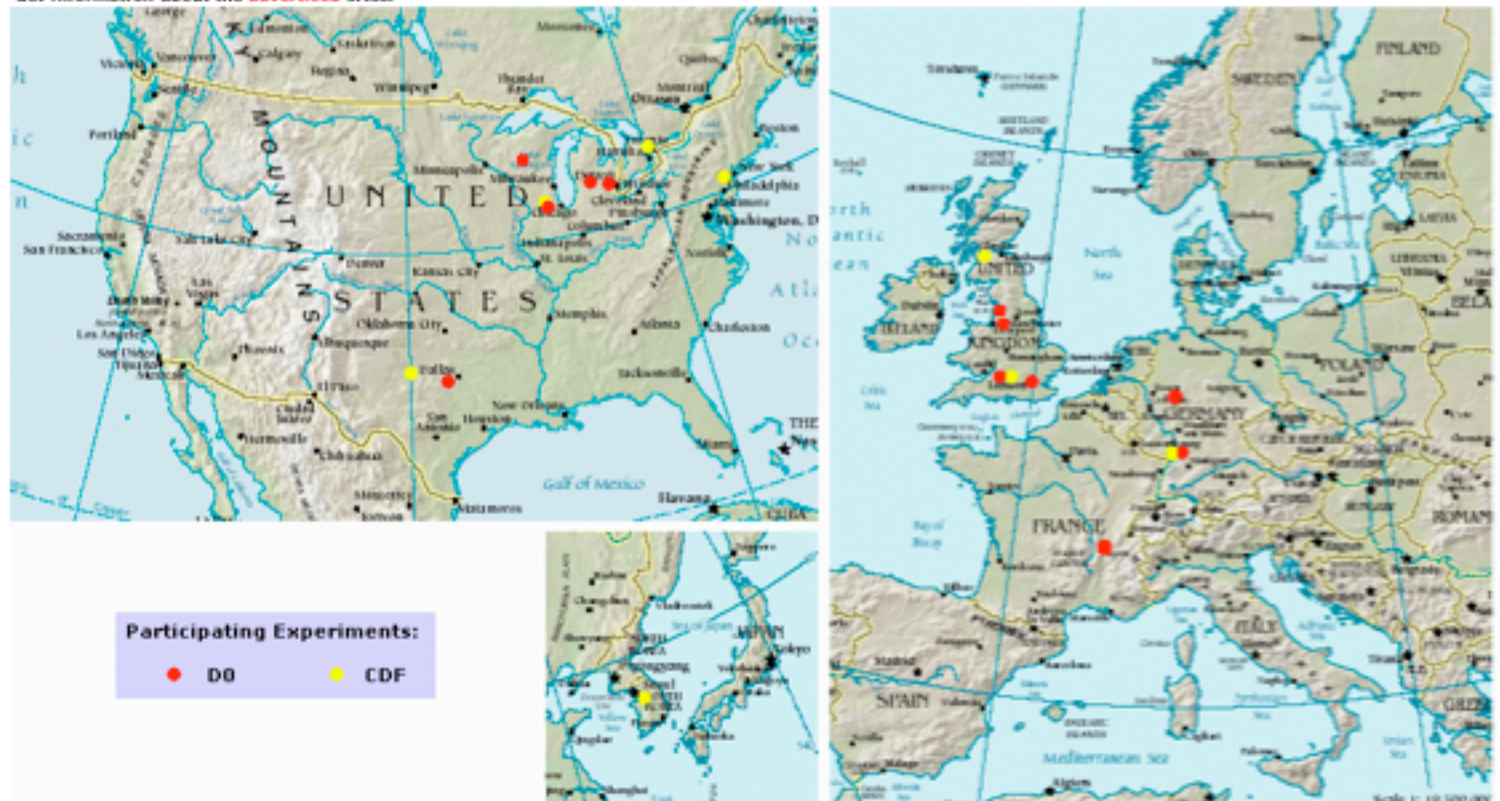
SAM GRID INFORMATION & MONITORING SYSTEM

Launching the Monitoring System:

Please click at the map to monitor the execution sites.
Get information about the **submission** sites.
Get information about the **advertised** sites.

◆ XMLDB

◆ Users can check on job progress



This web portal is best viewed with a 1280 x 1024 or higher screen resolution.

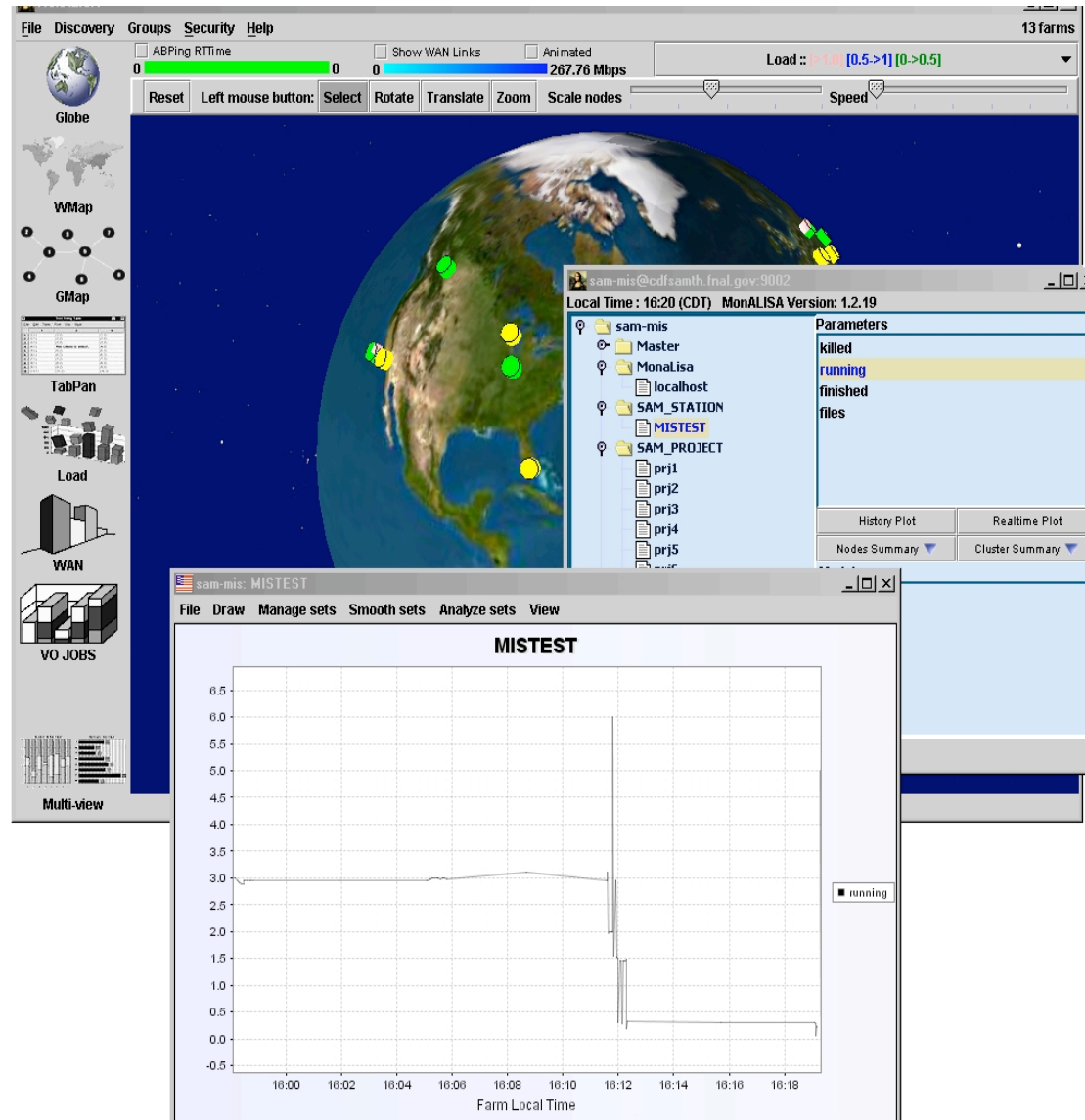
Future of Monitoring

- ◆ Current SamTV parses log files
 - ❖ Fragile, hard to maintain

2nd generation monitoring in the works

- ◆ Monitoring and Information Service (MIS)
 - ❖ MIS server receives events from SamGrid services via Corba (new project, open new file, delete file from cache) or can pull information from service
 - ❖ MIS Backends process events: store in local DB, send alert e-mail, update real time displays, export to other monitoring systems (MonaLisa)

SamGrid + MonaLisa



Test Harness

◆ Test Harness

❖ Unit testing of services is not enough

- Must mimic loads of a production system

❖ Performance and stress testing

- Discover problems, optimize performance

◆ Use a dedicated farm with SamGrid Test Harness to load the system

❖ Automatic tests with pass fail reports

- Check configuration of new installations

❖ Stress the system and use monitoring for results

Future of SamGrid

◆ Continuously refining our system

- ❖ Adapting to needs of other experiments
 - Minos has two detectors
- ❖ Refactoring and improving the implementation

◆ Adapting further to standard Grid tools

- ❖ Writing SamGrid SRM interfaces to access grid storage elements
- ❖ Interface to standard monitoring tools (but we need our own specific ones too)

- ❖ Moving to use of standard VO authorization

◆ Open problems

- ❖ More advanced brokering algorithms and scheduling
- ❖ VO Management - assign roles and attributes to users; finer grained security, temporary special privileges
- ❖ Automatically resubmit failed jobs (must be careful)

Summary

- ◆ SamGrid is a large scale distributed system integrating data delivery and job management for the many Petabyte data size era
- ◆ Successfully being used at DØ and CDF, initial deployment for MINOS. US-CMS investigating
- ◆ SamGrid continues to move into the Grid era

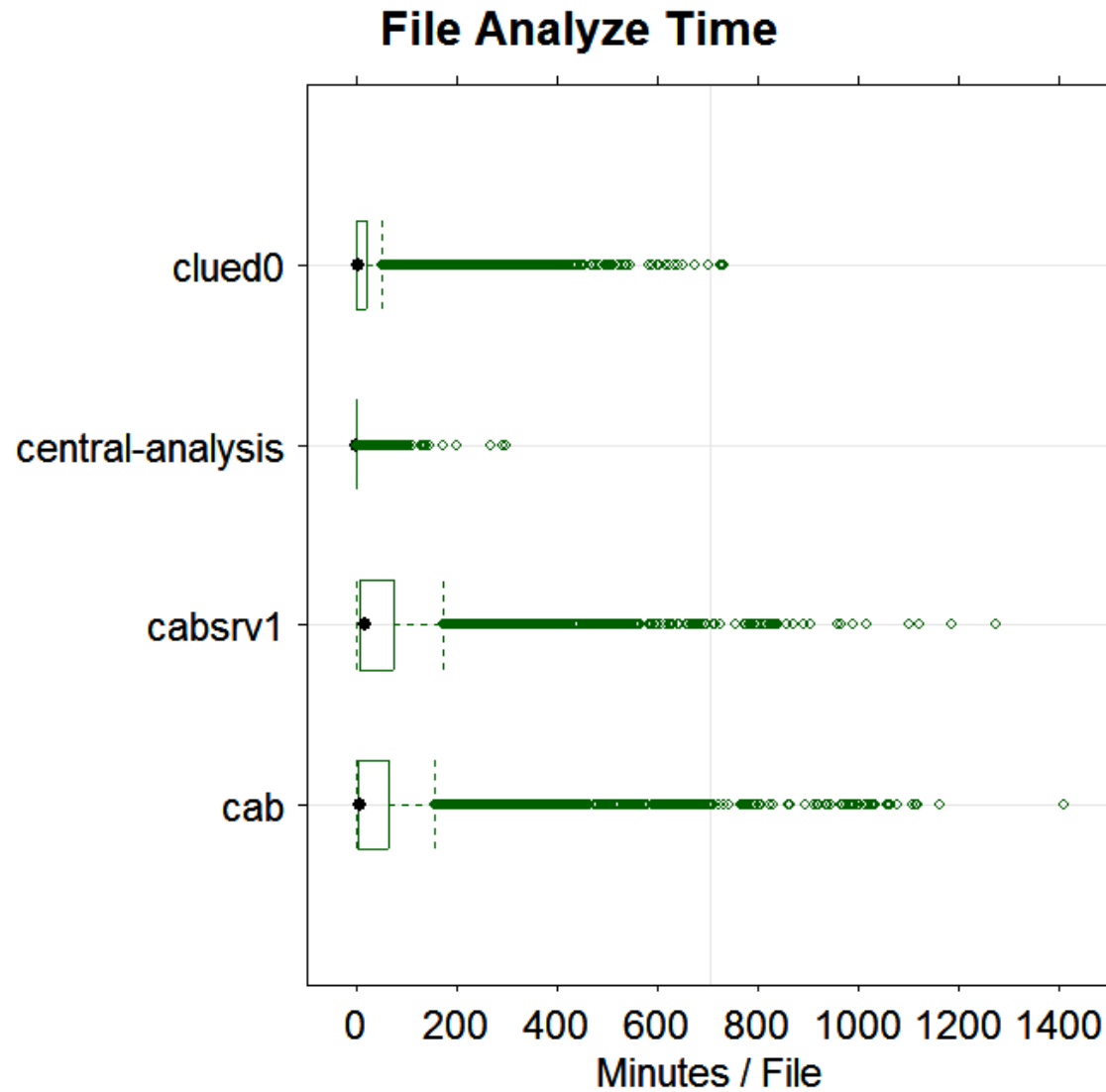
EXTRA SLIDES

◆ Extra slides go here

V. Re-reconstruction

- ◆ Reprocessing group submits projects to SamGrid. SamGrid chooses execution site and launches job(s)
- ◆ Jobs are run using *RunJob*, a work flow management system (CMS & DØ)
- ◆ Code arrives to job(s) via SamGrid
- ◆ Data arrives to job(s) via SamGrid
- ◆ Output files are sent back to FNAL for merging and storage back into SamGrid (future - will do on remote site)

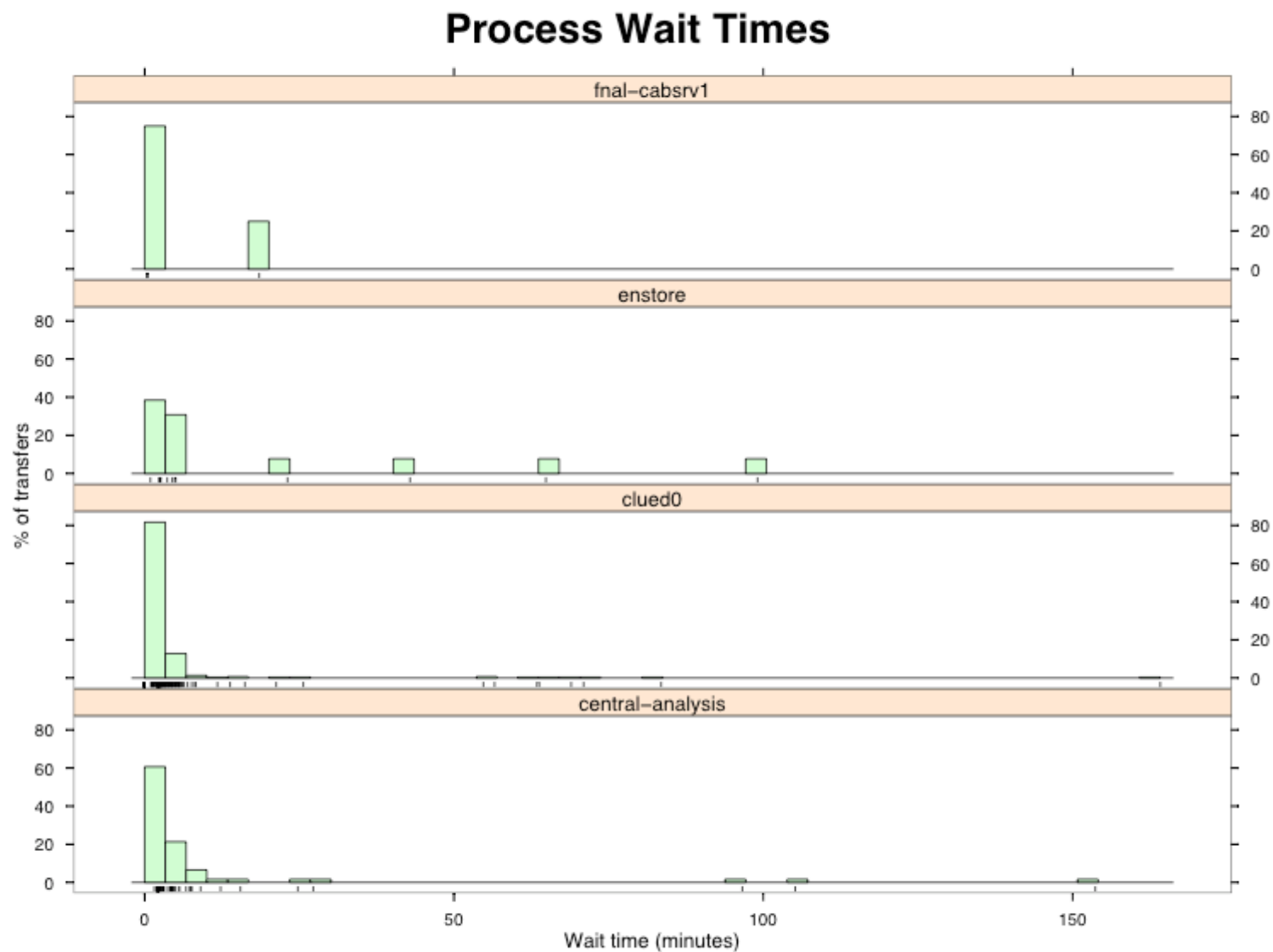
Process Execution Times



Failures

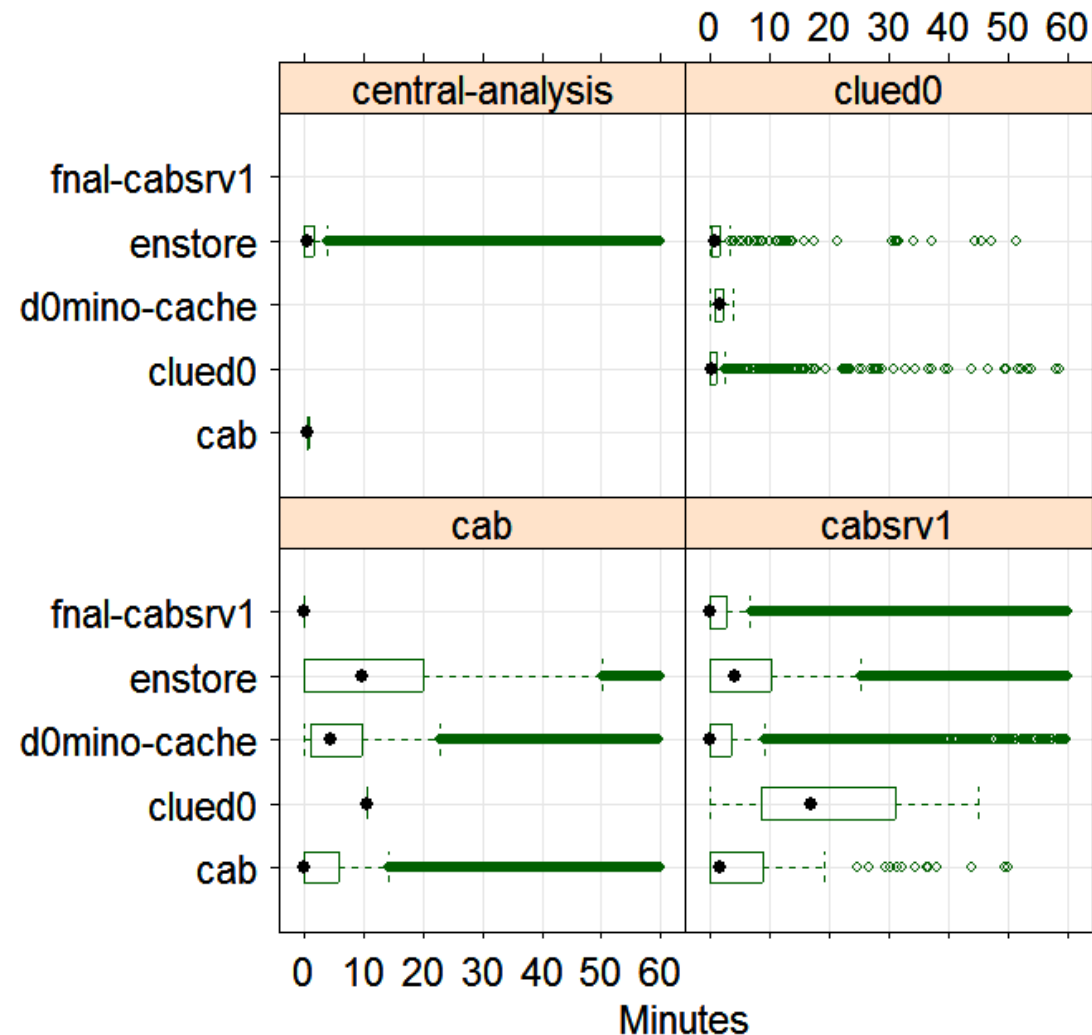
- ◆ On linux nodes, $\sim 1\%$ files are not successfully consumed
 - ❖ Application crashes (pilot error)
 - ❖ IDE disk problems (must check CRC after each file transfer)
 - ❖ Hardware failures
 - ❖ Temporary no access to certain tapes
- ◆ On SMP machine, failure rate is 0.1%
 - ❖ Hardware and disks are much more robust
 - ❖ People tend to run standard applications

SamTV History



Process Wait Times

Wait Time for File Delivery (truncated)



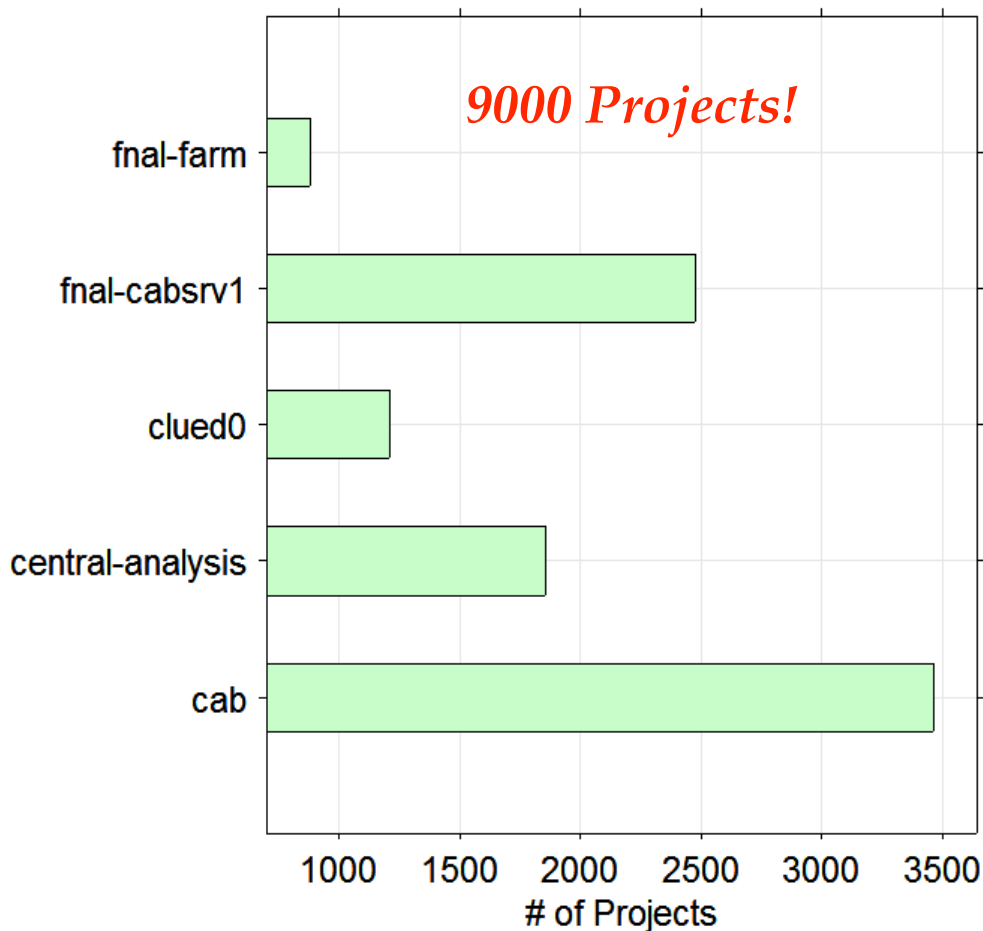
- ◆ Time between *Request Next File* and *Open File*
- ◆ For CAB and CABSRV1
 - ❖ 50% of enstore transfers occur within 10 minutes.
 - ❖ 75% within 20 minutes
 - ❖ 95% within 1 hour
- ◆ For CENTRAL-ANALYSIS and CLUED0
 - ❖ 95% of enstore transfers within 10 minutes

Station	CAB	CABSRV1
% no wait	30%	40%

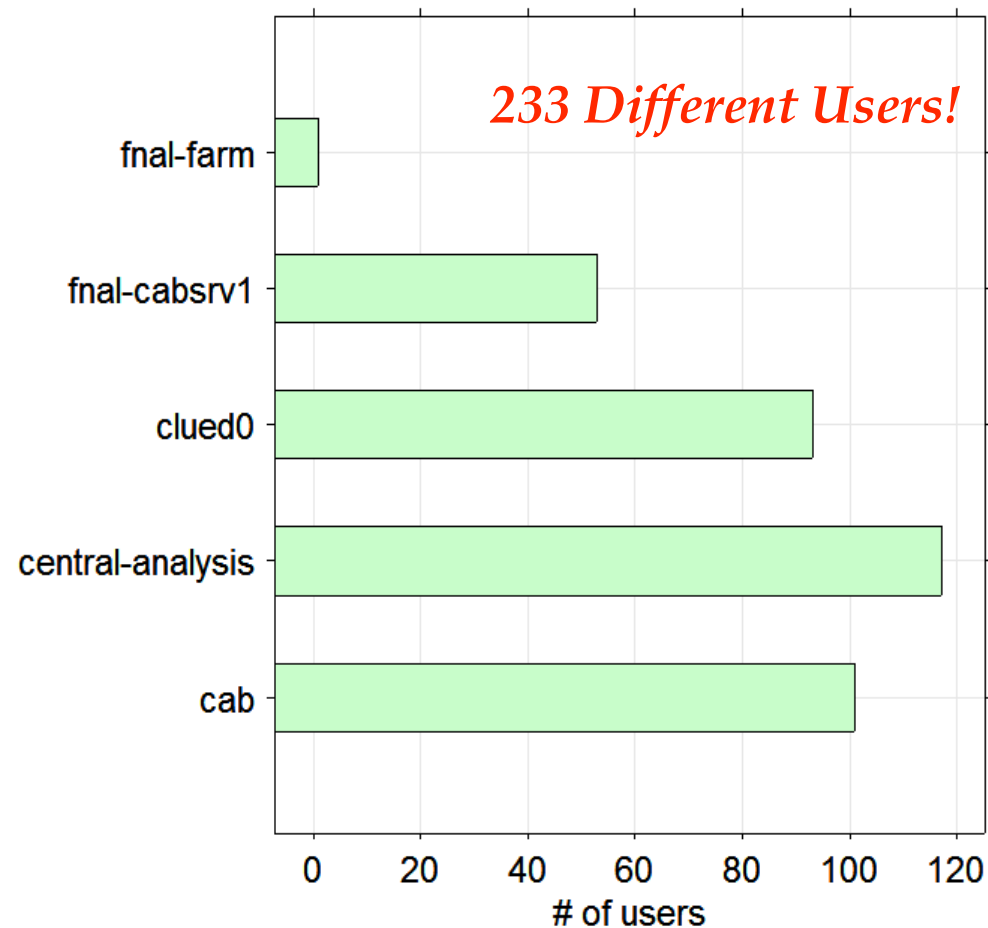
SAMGrid Statistics - Usage Data

Data from early January 6 until February 24 at DØ

of Projects Started

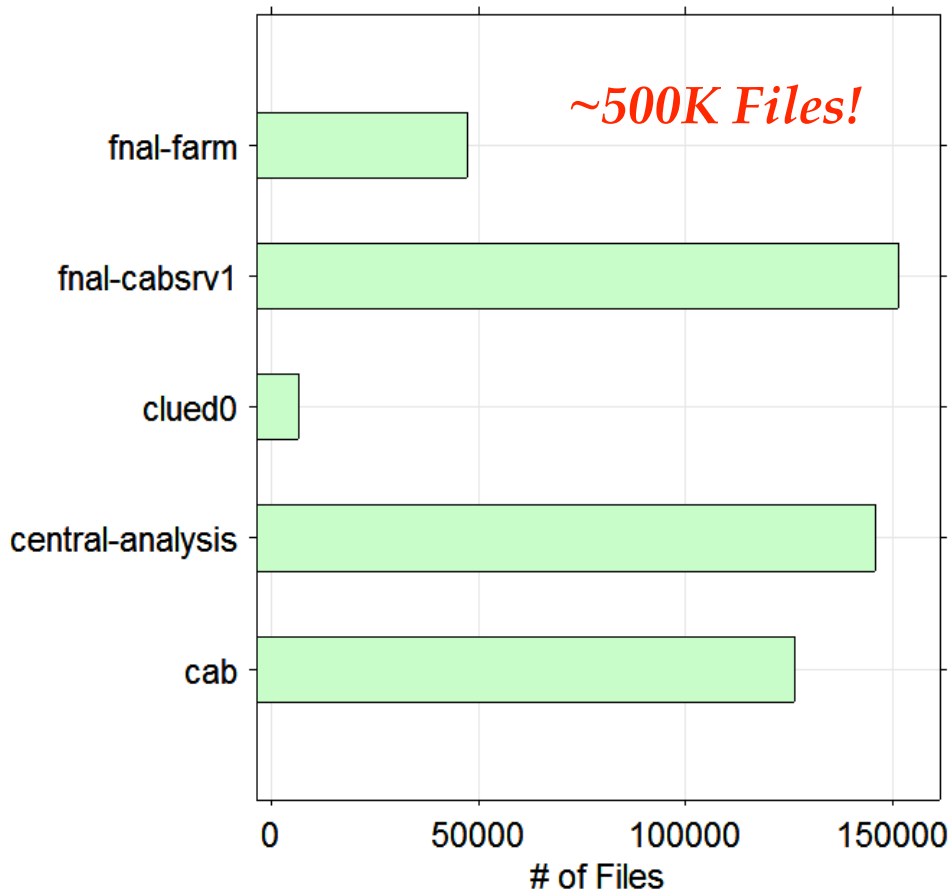


of Different Users

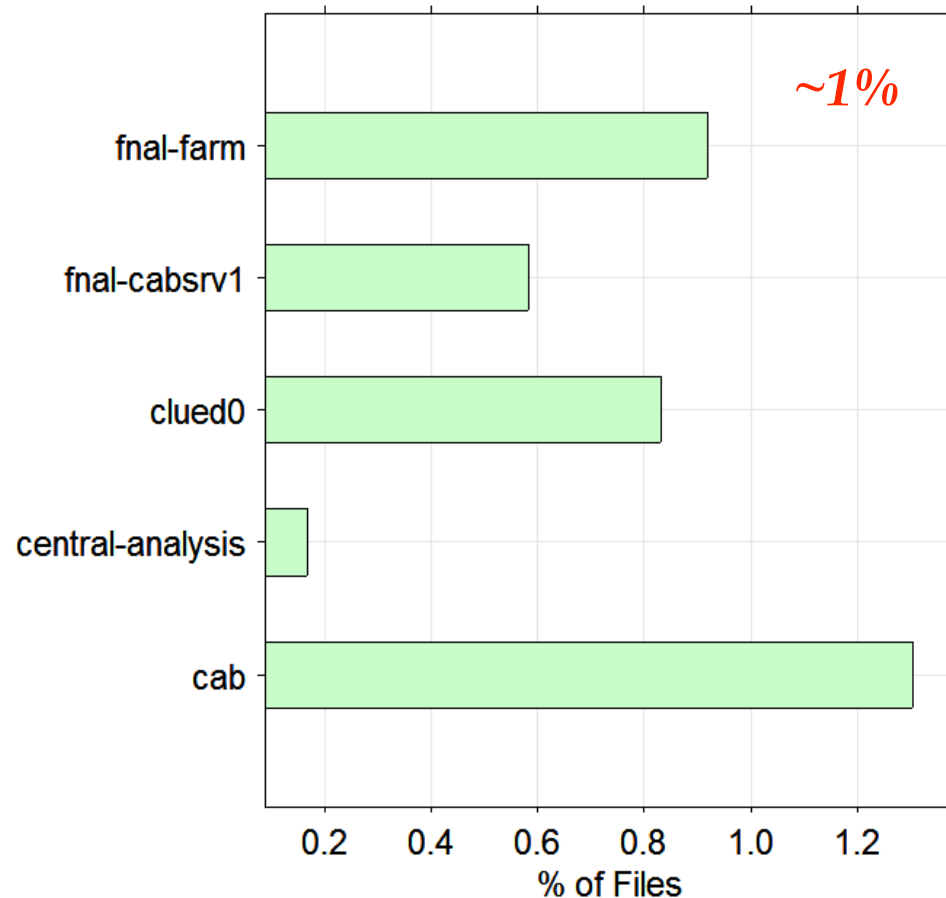


SAMGrid Statistics - Usage Data

of Consumed Files



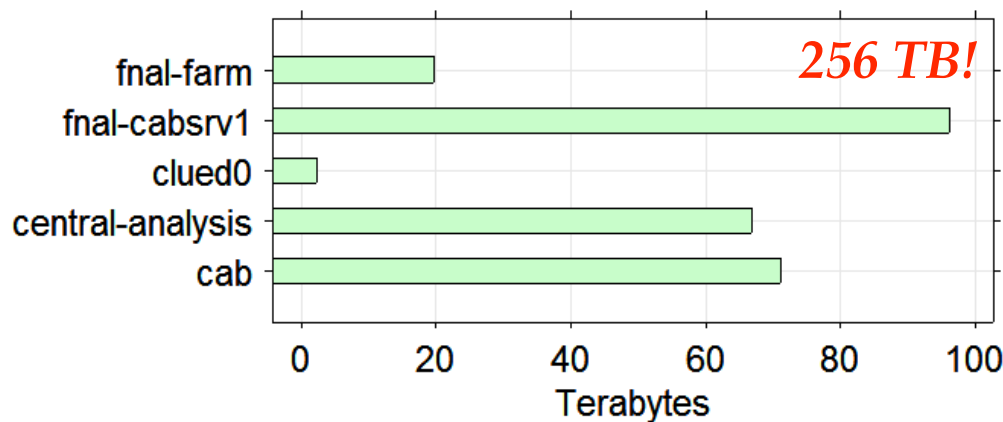
% of Files Unconsumed



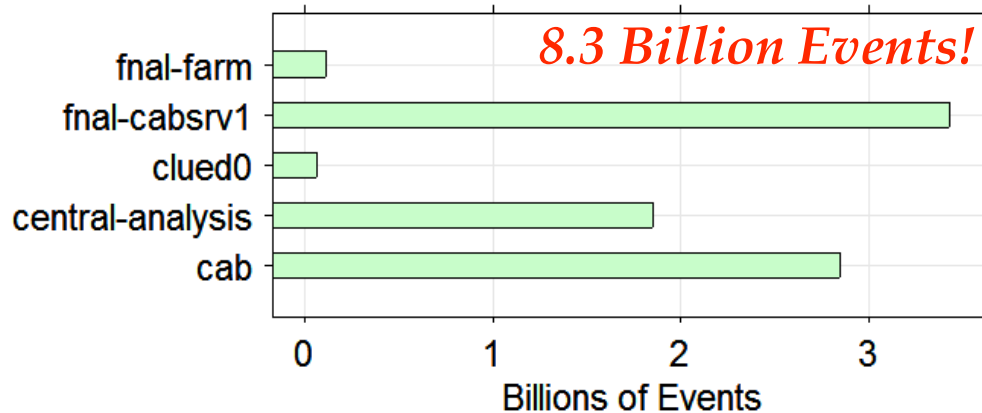
SAMGrid Statistics - Usage Data

Data from early January 6 until February 24 at DØ

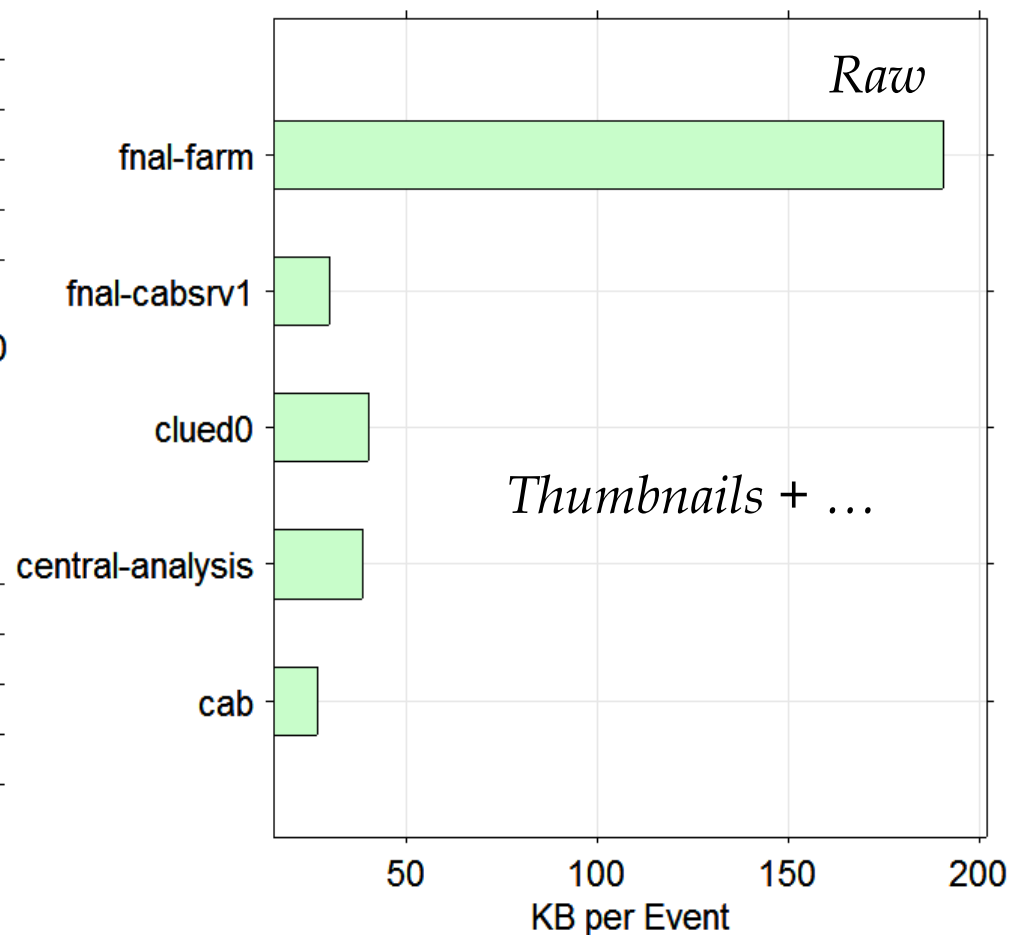
Terabytes Consumed



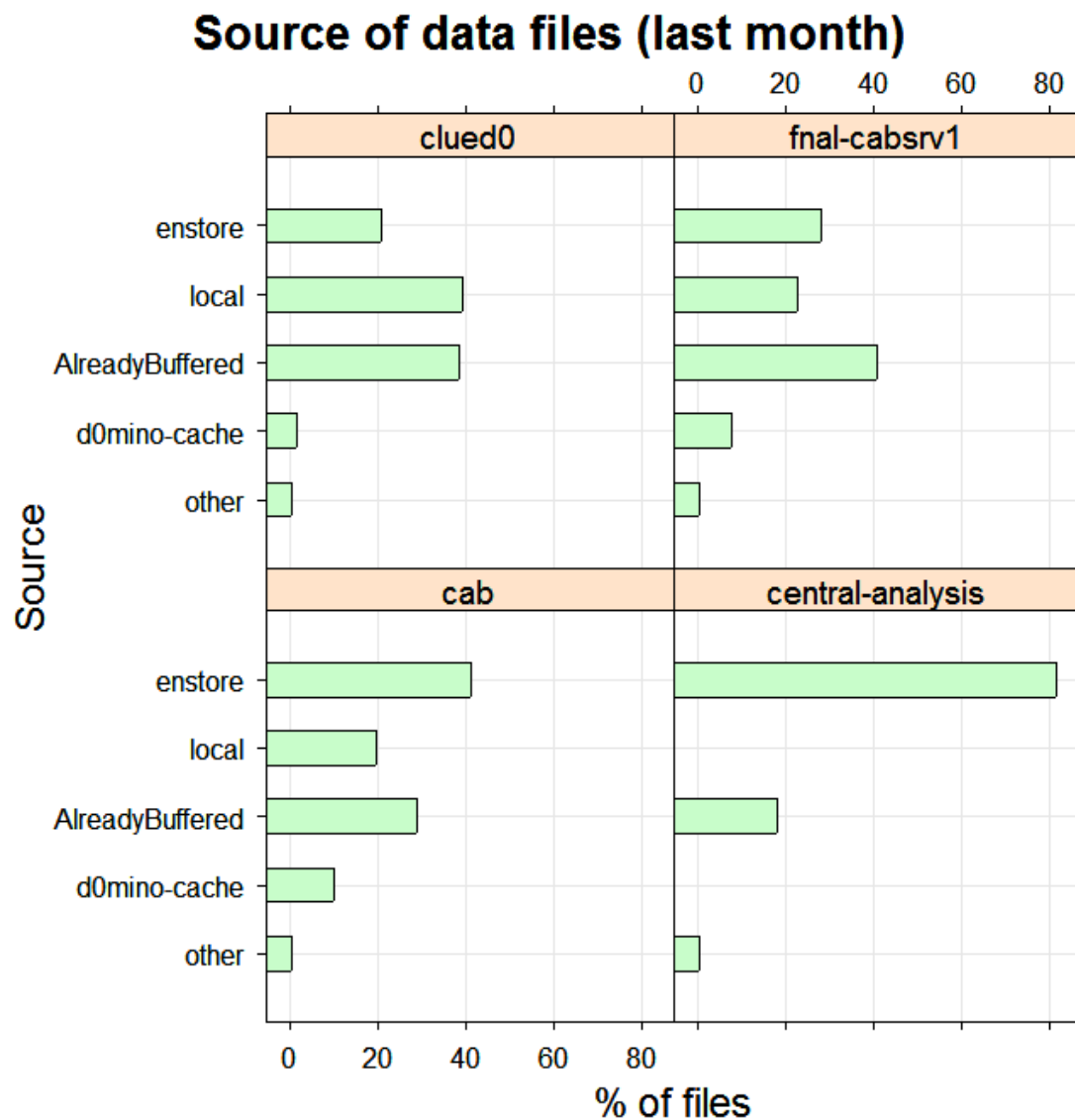
Billions of Events served



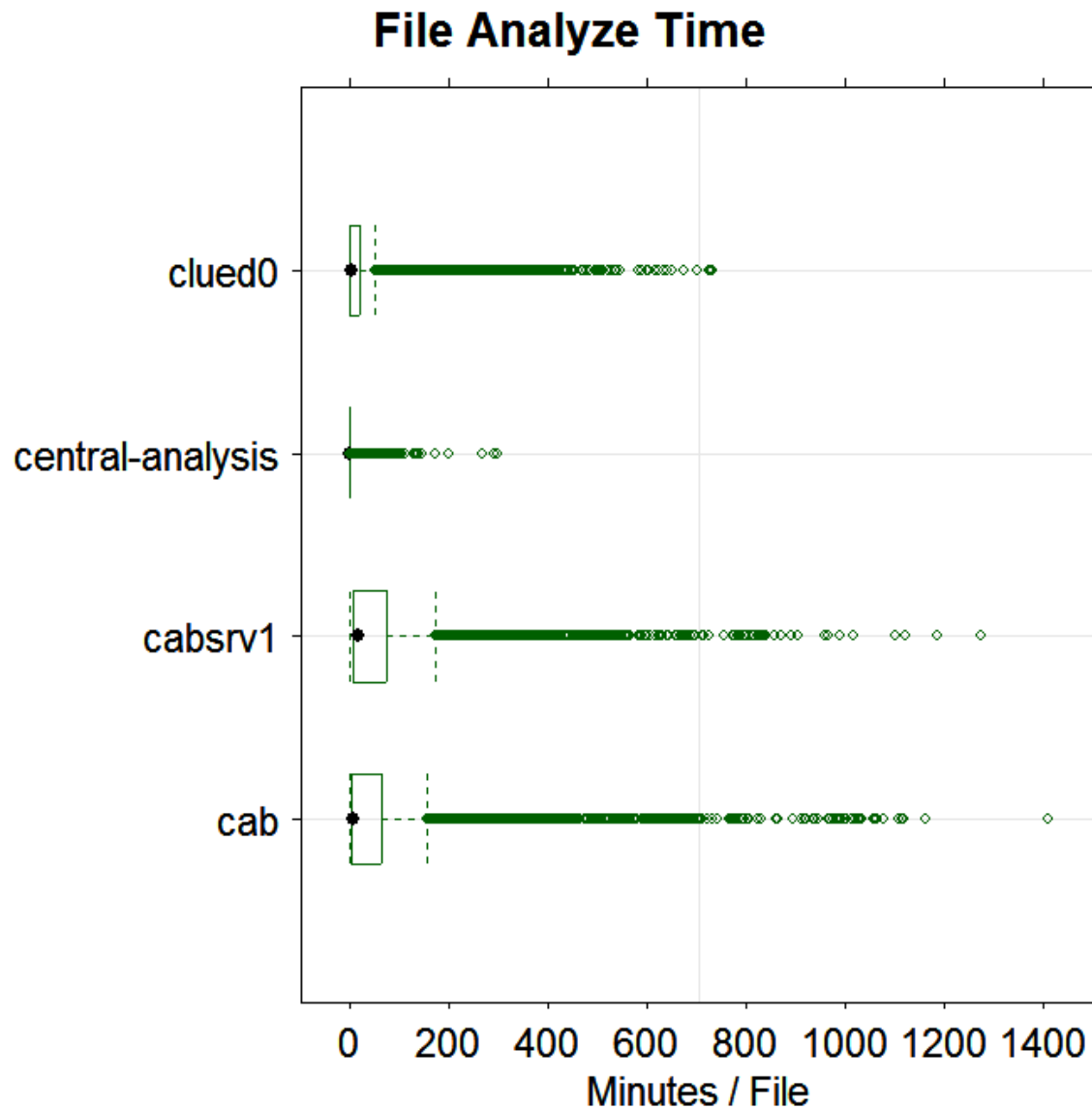
Average Event Size



SAMGrid Statistics - Operations Data



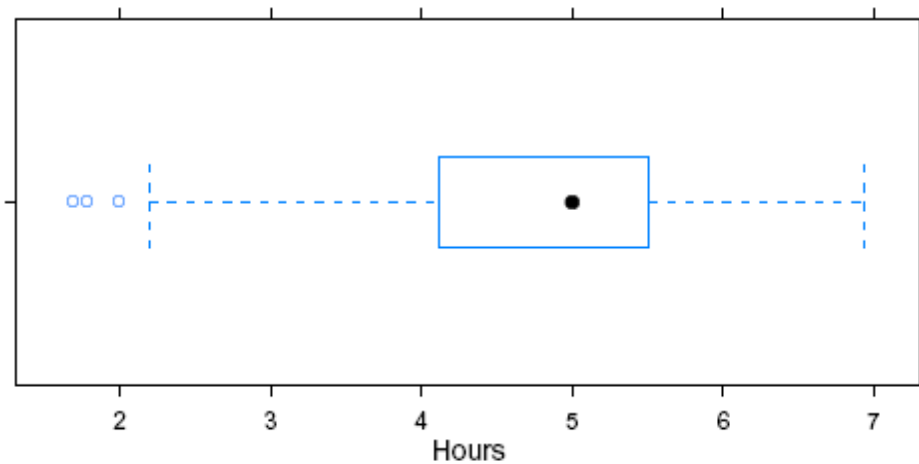
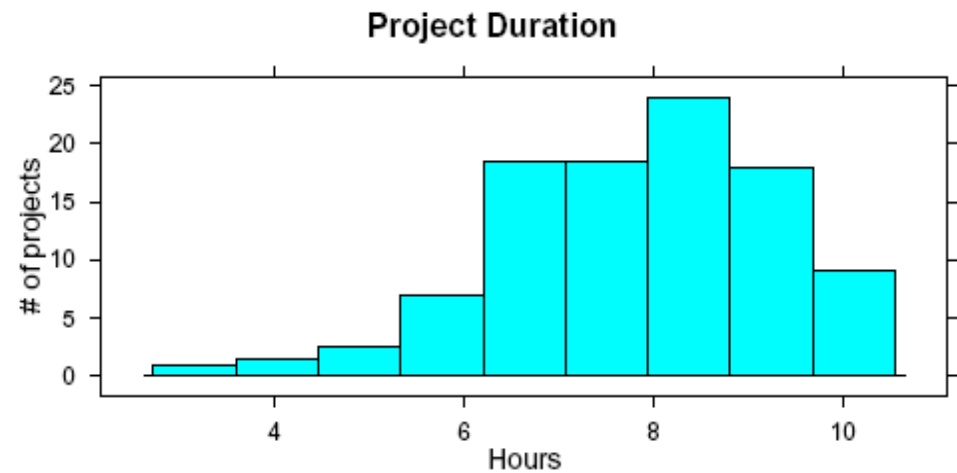
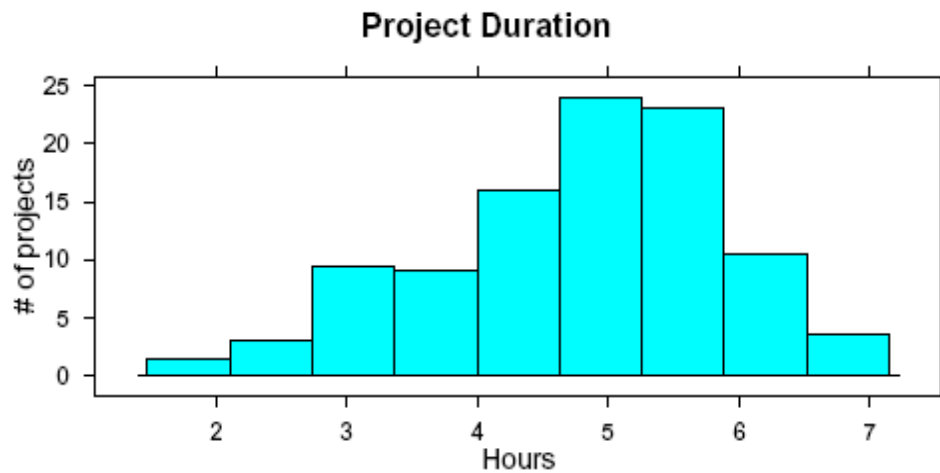
SAMGrid Statistics - Operations Data



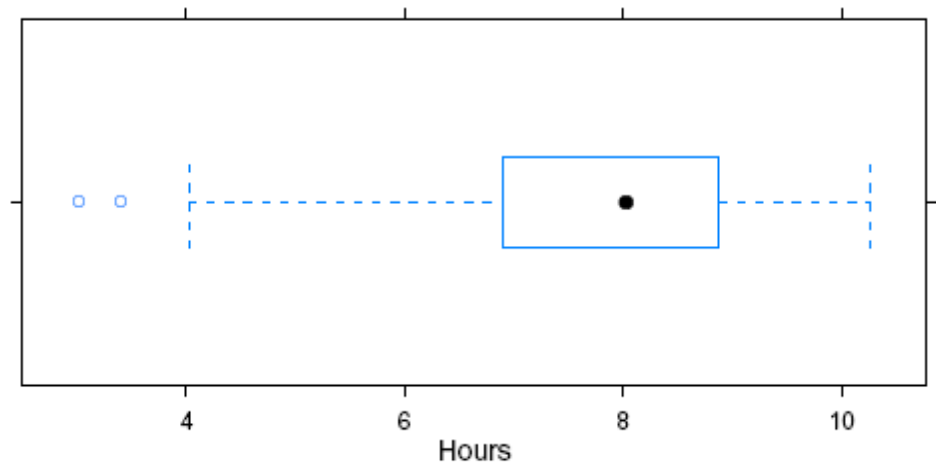
Stress Testing

- ◆ There are many station parameters to tune
 - ❖ Maximum parallel transfers
 - ❖ Maximum concurrent enstore requests
 - ❖ Configuration of cache disks
 - ❖ ...
- ◆ We're moving away from d0mino to Linux
 - ❖ How robust are these linux machines?
 - ❖ How many projects can they run?
 - ❖ How many concurrent file transfers can they handle?
- ◆ Running test harness on a small cluster to explore SAMGrid parameter space

SAMGrid Stress Testing



max transfers =5

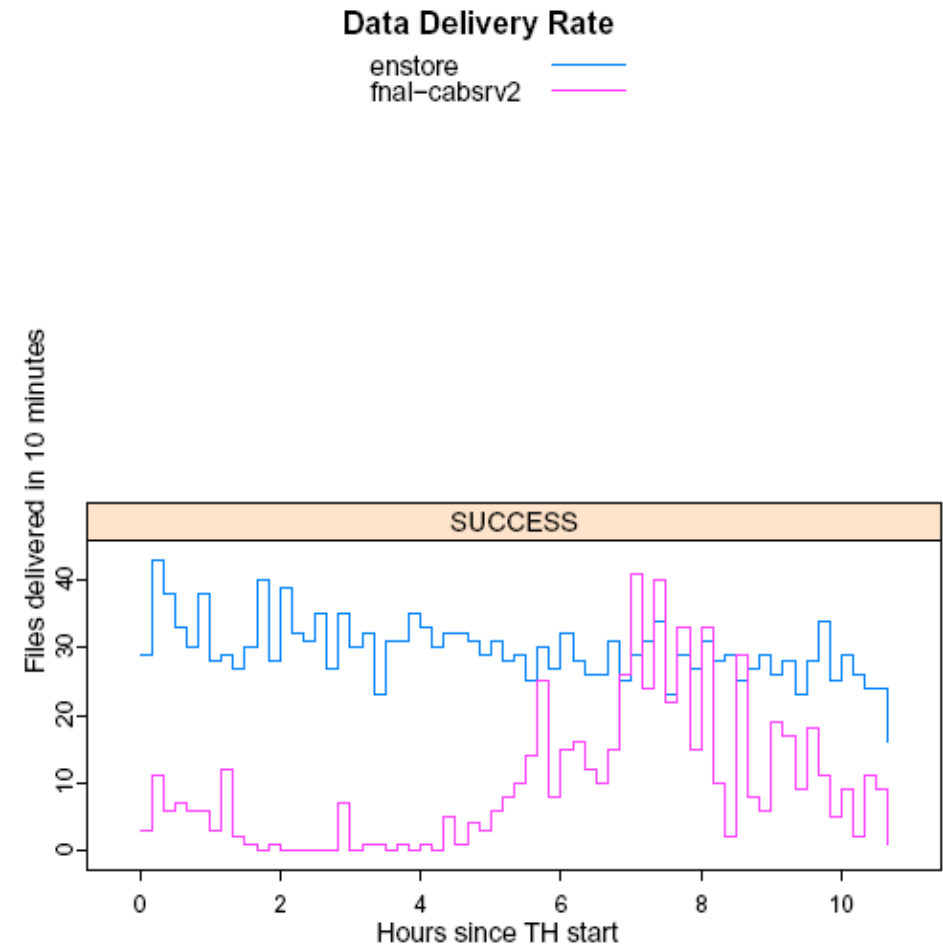


max transfers =1

SAMGrid Stress Testing

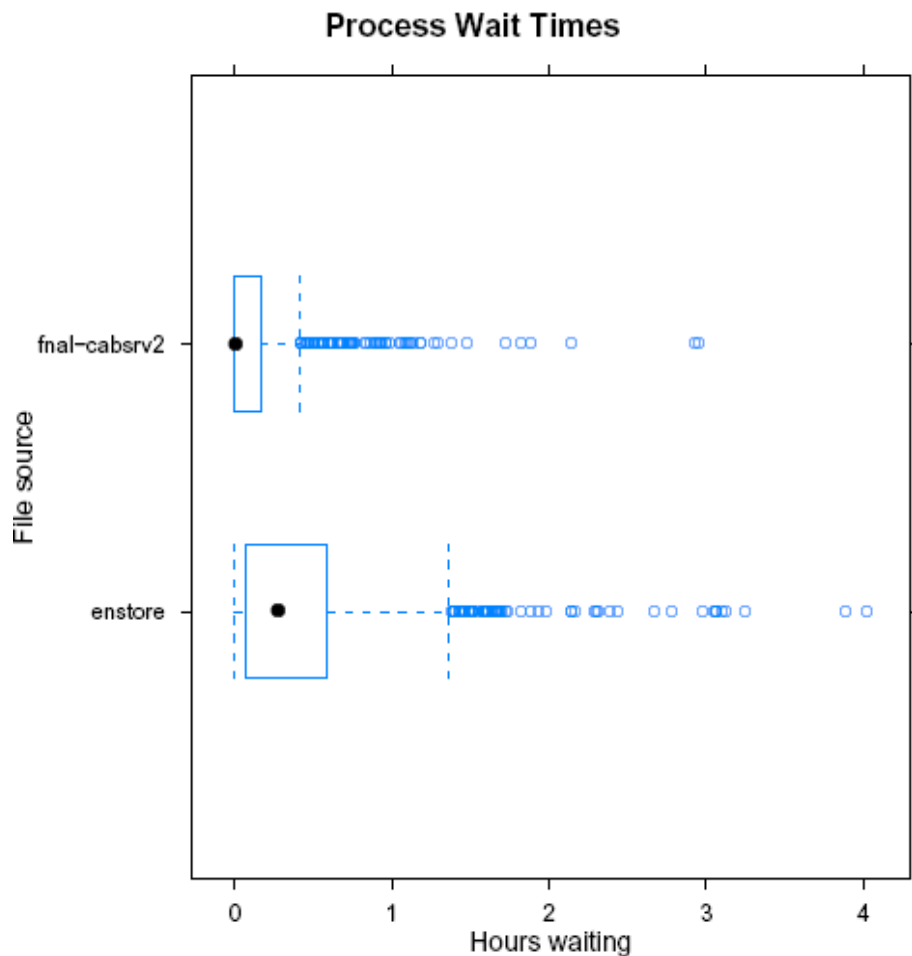


max transfers =5

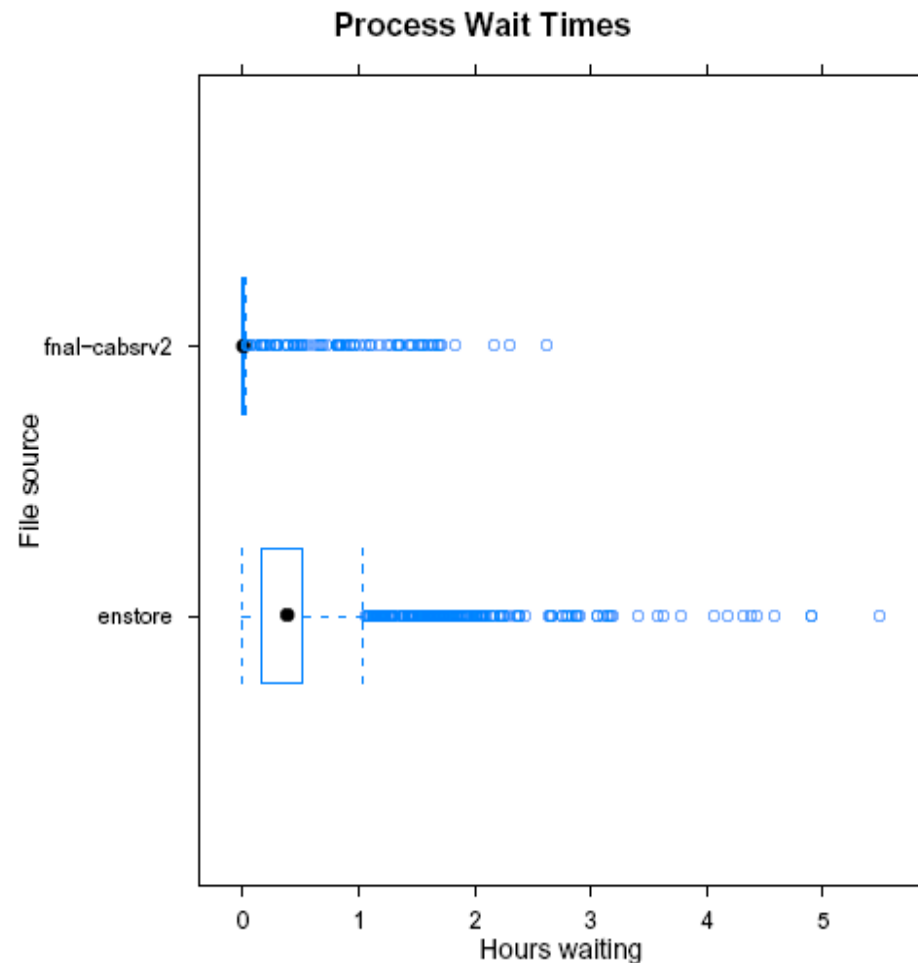


max transfers =1

SAMGrid Stress Testing



max transfers =5

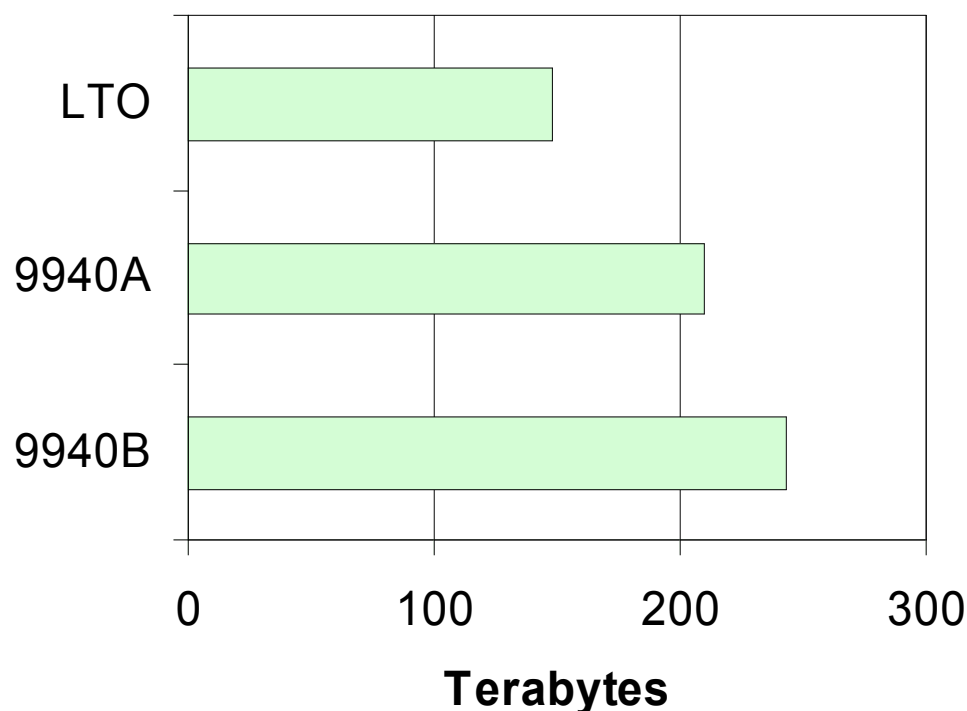


max transfers =1

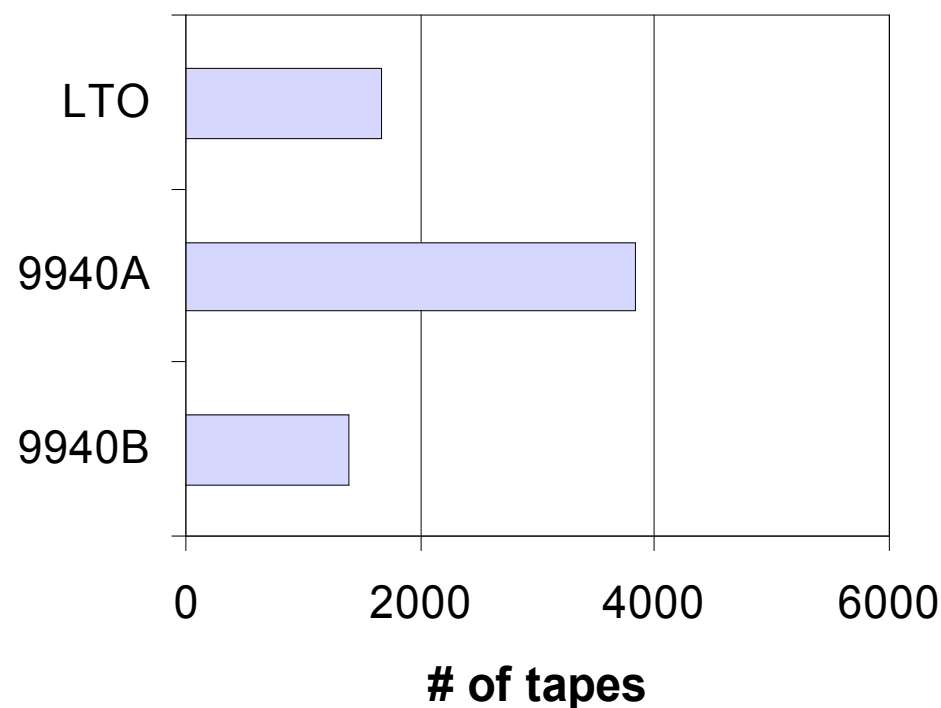
ENSTORE Statistics

◆ 0.6 Petabytes in tape storage!

Data sizes



Tape usage

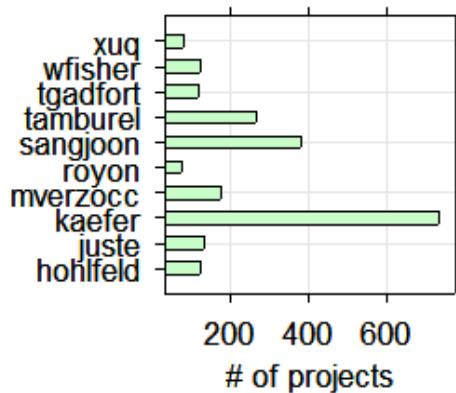


Only 5 files unrecoverable (5 GB total; 8ppm loss) !!!
One of them was RAW file

Top Users (Jan 6, 2004 - Feb 24, 2004)

Top users by # of projects

Top users on cab

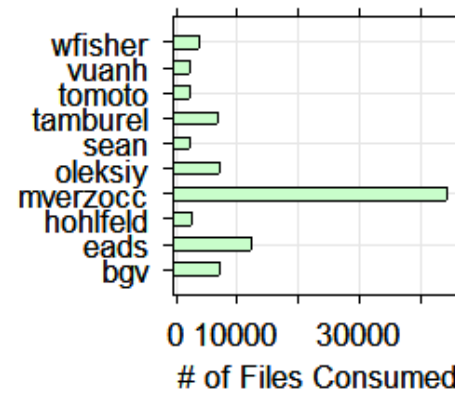


Top users on fnal-cabsrv1

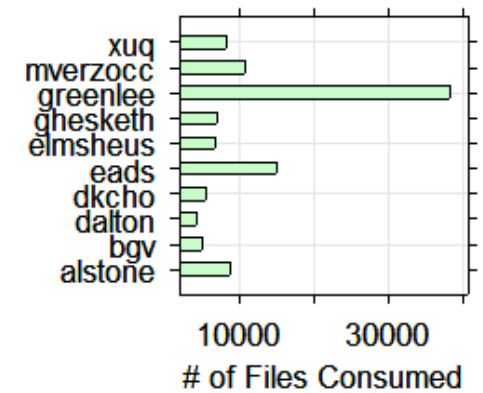


Top users by consumed files

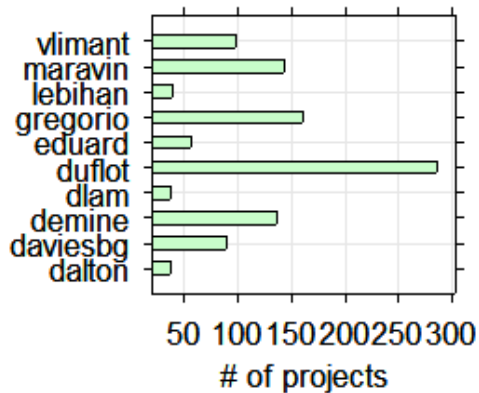
Top users on cab



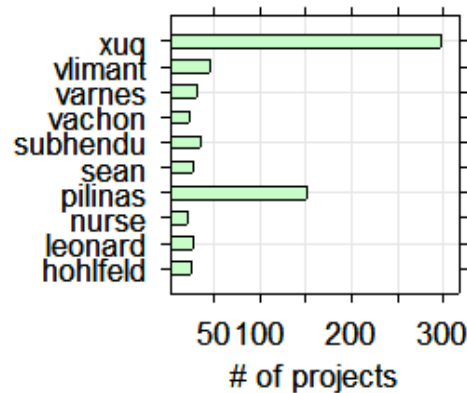
Top users on fnal-cabsrv1



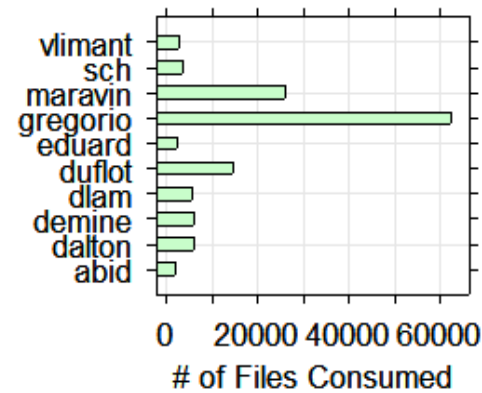
Top users on central-analysis



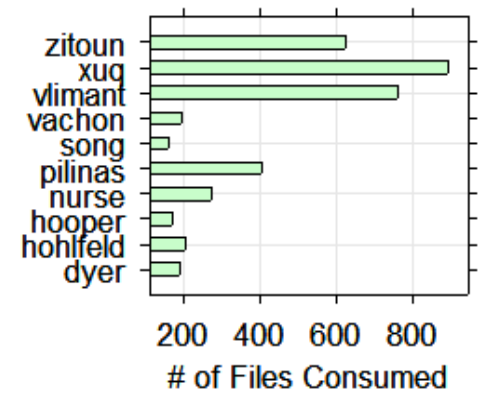
Top users on clued0



Top users on central-analysis



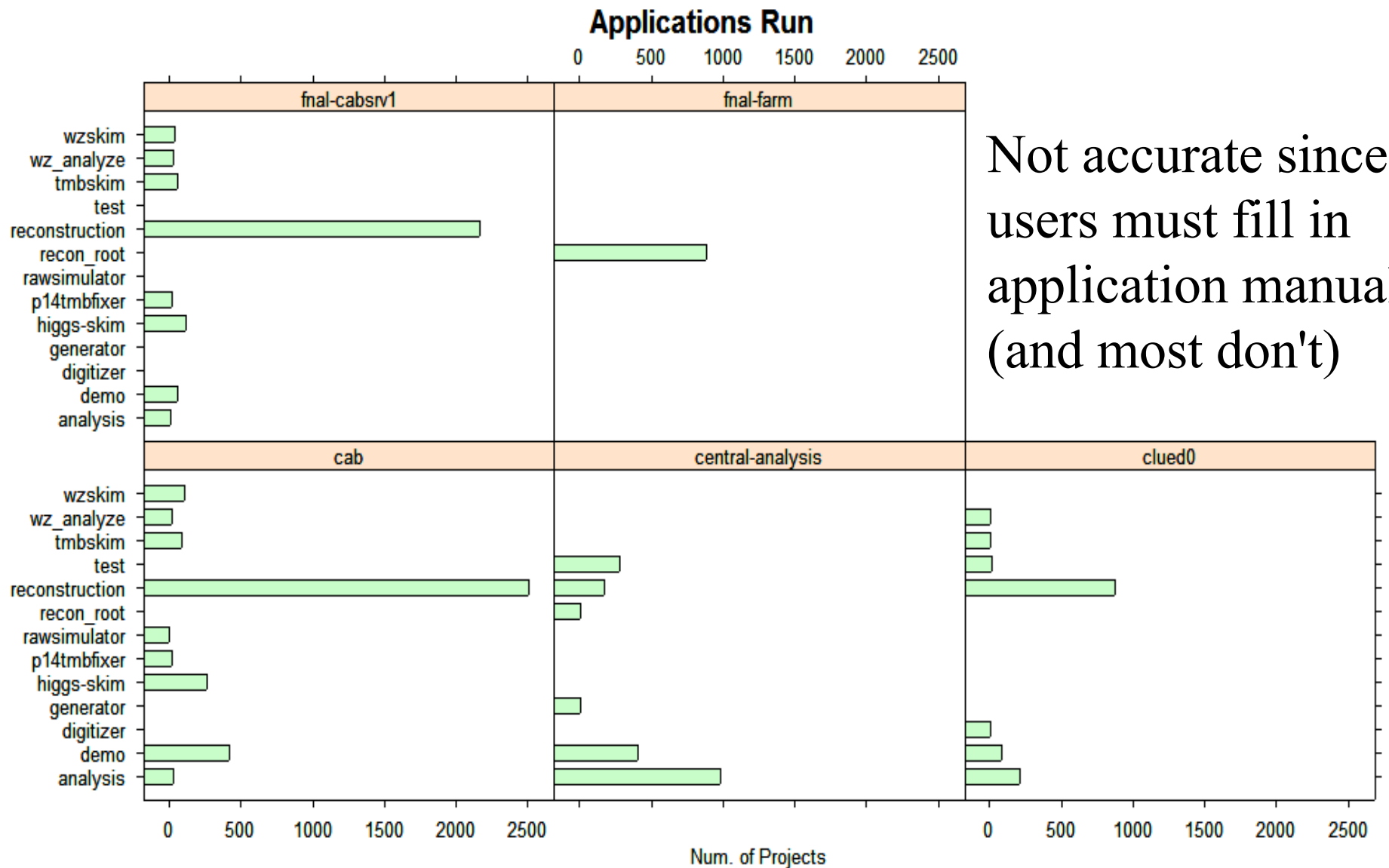
Top users on clued0



SAMGrid Statistics

people doing?

What are

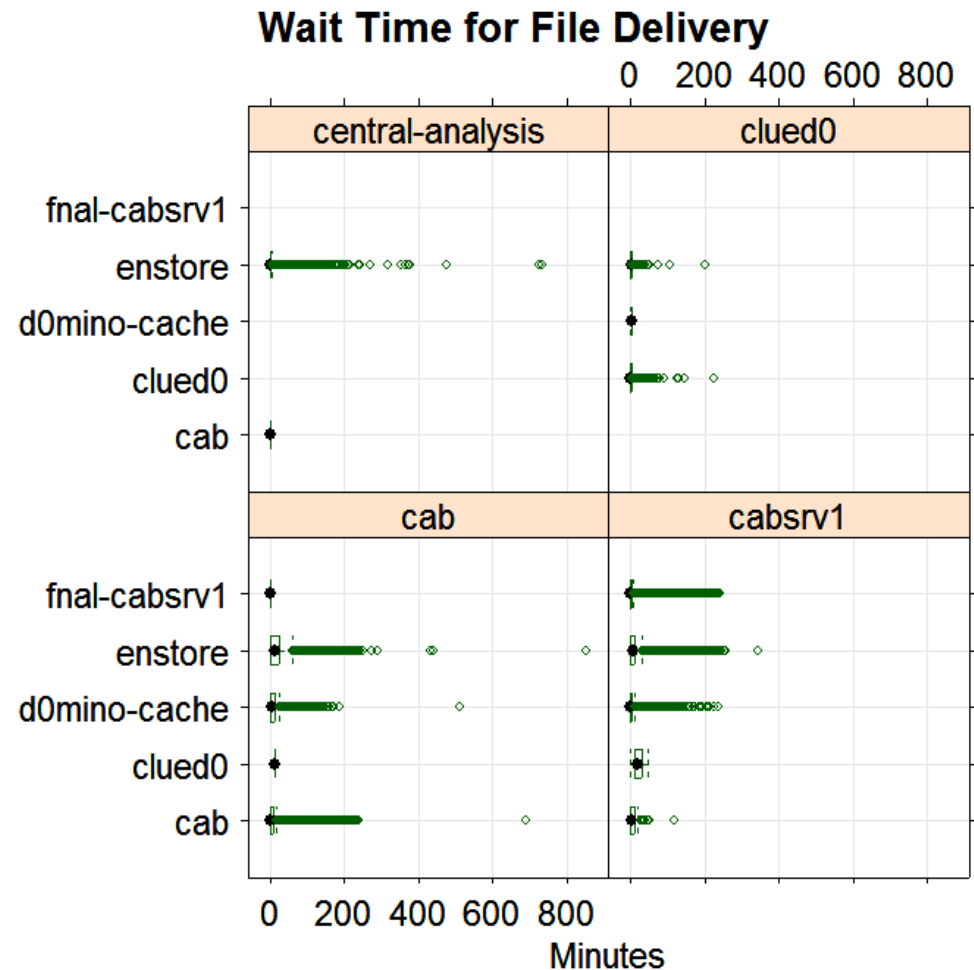
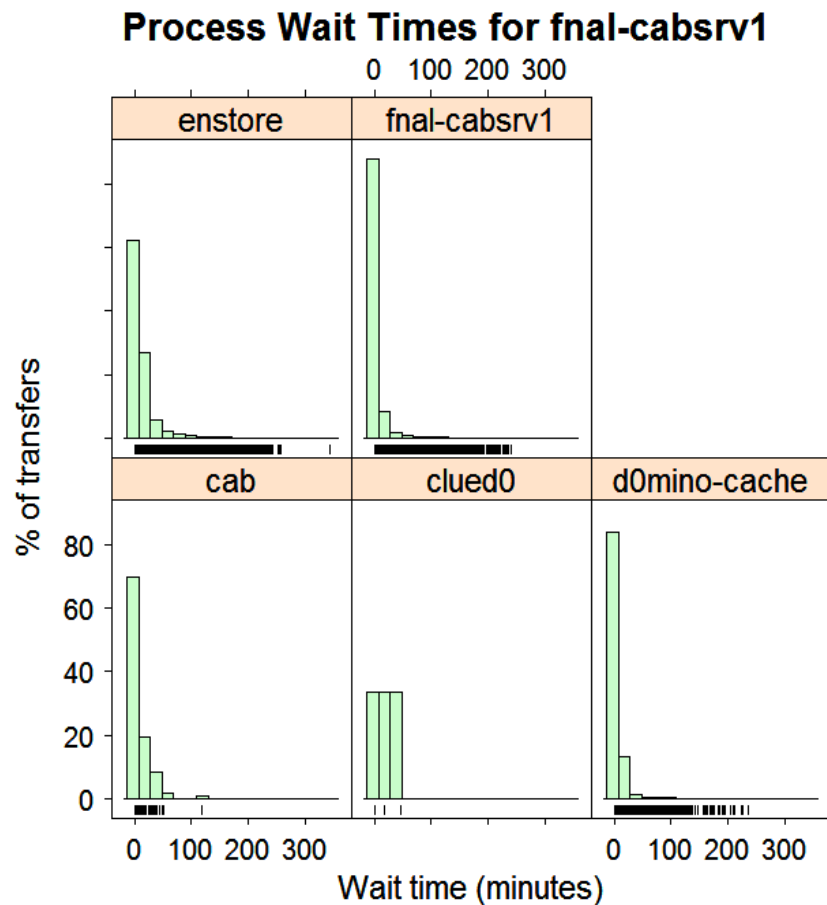


SAMGrid Statistics

Process

wait times

File Source



Some SAMGrid buzzwords

◆ Dataset Definition

- ❖ A set of requirements to obtain a particular set of files
- ❖ *e.g.* data_tier thumbnail and run_number 181933
- ❖ Datasets can change over time
 - More files that satisfy the dataset may be added to SAMGrid

◆ Snapshot

- ❖ The files that satisfy a dataset at a particular time (e.g. when you start an analysis job)
- ❖ Snapshots are static

◆ Project

- ❖ The running of an executable over files in SAMGrid
- ❖ Consists of the dataset definition, the snapshot from that dataset definition, and application information
- ❖ Bookkeeping data is kept - how many files did you successfully process, where did your job run, how long did it take

SAM-GRID Projects

- ◆ **Active Subprojects:** C++ API, DBServer, JIM, H Stream Reco for CDF, Caching, Chains&Links, CDF DFC, Test Harness, Linux deploy of DBServers, Config Man
- ◆ **Planned Subprojects:** Request system, Autodest, Further monitoring (MIS)
- ◆ **Related Subprojects:** d0tools, SBIR II, Condor mods, workflow packages for CDF & D0, Authorization & Accounting
- ◆ **Recently completed Subprojects:** Python API, V5.1 Schema Design, Batch Adapter, D0 Online dcache TDP, 1st Gen Monitoring Tools, Data Dimensions Grammar

DB Servers

